1992-09

NPSNET: aural cues for virtual world immersion

Dahl, Leif Alan

Monterey, California. Naval Postgraduate School

http://hdl.handle.net/10945/23731

SECURITY CLASSIFICATION OF THIS PAGE

REPORT DOCUMENTATION PAGE

1a. REPORT SECURITY CLASSIFICATION UNCLASSIFIED	1b. RESTRICTIVE MARKINGS
2a. SECURITY CLASSIFICATION AUTHORITY 2b. DECLASSIFICATION/DOWNGRADING SCHEDULE	3. DISTRIBUTION/AVAILABILITY OF REPORT Approved for public release; distribution is unlimited
4. PERFORMING ORGANIZATION REPORT NUMBER(S)	5. MONITORING ORGANIZATION REPORT NUMBER(S)
6a. NAME OF PERFORMING ORGANIZATION Computer Science Dept. Naval Postgraduate School 6b. OFFICE SYMBOL (if applicable) CS/ZK	7a. NAME OF MONITORING ORGANIZATION Naval Postgraduate School
6c. ADDRESS (City, State, and ZIP Code) Monterey, CA 93943-5000	7b. ADDRESS (City, State, and ZIP Code) Monterey, CA 93943-5000
8a. NAME OF FUNDING/SPONSORING ORGANIZATION 8b. OFFICE SYMBOL (if applicable)	9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER
8c. ADDRESS (City, State, and ZIP Code)	10. SOURCE OF FUNDING NUMBERS PROGRAM ELEMENT NO. / PROJECT NO. / TASK NO. / WORK UNIT ACCESSION N

11. TITLE (Include Security Classification)
NPSNET: AURAL CUES FOR VIRTUAL WORLD IMMERSION

12. PERSONAL AUTHOR(S)
Dahl, Leif A.

13a. TYPE OF REPORT	13b. TIME COVERED	14. DATE OF REPORT (Year, Month, Day)	15. PAGE COUNT
Master's Thesis	FROM 04/90 TO: 09/92	September 1992	75

16. SUPPLEMENTARY NOTATION The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the United States Government.

17. COSATI CODES			18. SUBJECT TERMS (Continue on reverse if necessary and identify by block number)
FIELD	GROUP	SUB-GROUP	Graphics, Simulators, Sound, MIDI, Interface, Macintosh, Sampler

19. ABSTRACT (Continue on reverse if necessary and identify by block number)

NPSNET is a low-cost visual and aural simulation system designed and implemented at the Naval Postgraduat School. NPSNET is an example of a virtual world simulation environment that incorporates real-time aural cue through software-hardware interaction. In the current implementation of NPSNET, a graphics workstation function in the sound server role which involves sending and receiving networked sound message packets across a Local Are Network, composed of multiple graphics workstations. The network messages contain sound file identificatio information that is transmitted from the sound server across an RS-422 protocol communication line to a serial t Musical Instrument Digital Interface (MIDI) converter. The MIDI converter, in turn relays the sound byte to sampler, an electronic recording and playback device. The sampler correlates the hexadecimal input to a specific not or stored sound and sends it as an audio signal to speakers via an amplifier. The realism of a simulation is improve by involving multiple participant senses and removing external distractions. This thesis describes the incorporatio of sound as aural cues, and the enhancement they provide in the virtual simulation environment of NPSNET.

20. DISTRIBUTION/AVAILABILITY OF ABSTRACT [X] UNCLASSIFIED/UNLIMITED [ ] SAME AS RPT. [ ] DTIC USERS	21. ABSTRACT SECURITY CLASSIFICATION UNCLASSIFIED	
22a. NAME OF RESPONSIBLE INDIVIDUAL Michael J. Zyda	22b. TELEPHONE (Include Area Code) (408) 646-2305	22c. OFFICE SYMBOL CS/ZK

DD FORM 1473, 84 MAR — 83 APR edition may be used until exhausted. All other editions are obsolete

SECURITY CLASSIFICATION OF THIS PAGE
UNCLASSIFIED

Approved for public release; distribution is unlimited

NPSNET: AURAL CUES FOR VIRTUAL WORLD IMMERSION

by
Leif Alan Dahl
Lieutenant, United States Navy
B.S., United States Naval Academy, 1984

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF COMPUTER SCIENCE

from the

NAVAL POSTGRADUATE SCHOOL
September 1992

ABSTRACT

NPSNET is a low-cost visual and aural simulation system designed and implemented at the Naval Postgraduate School. NPSNET is an example of a virtual world simulation environment that incorporates real-time aural cues through software-hardware interaction. In the current implementation of NPSNET, a graphics workstation functions in the sound server role which involves sending and receiving networked sound message packets across a Local Area Network, composed of multiple graphics workstations. The network messages contain sound file identification information that is transmitted from the sound server across an RS-422 protocol communication line to a serial to Musical Instrument Digital Interface (MIDI) converter. The MIDI converter, in turn relays the sound byte to a sampler, an electronic recording and playback device. The sampler correlates the hexadecimal input to a specific note or stored sound and sends it as an audio signal to speakers via an amplifier. The realism of a simulation is improved by involving multiple participant senses and removing external distractions. This thesis describes the incorporation of sound as aural cues, and the enhancement they provide in the virtual simulation environment of NPSNET.

c.1

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ACKNOWLEDGEMENTS

The incorporation of sound in NPSNET has been a difficult undertaking, primarily due to the many facets of computer science the project has encompassed. In a similar manner, many people have assisted in the development of these different facets and I would like to thank and acknowledge them here.

Susannah Bloch provided invaluable assistance in just about every area of this research. From her continually happy countenance and ever present smile, to her adeptness at programming in C, mastery of Unix and of the Emax II sound system, she has been a pleasure to work with. I thank you Susannah and wish you the best at Harvey Mudd College.

Dr. Michael Zyda and David Pratt are undoubtedly the best thesis advisor and co-advisor I could have asked for. From my first exposure to these two individuals, I have been motivated by their enthusiasm, diligence and expertise. Thesis research is much more enjoyable and motivating when one's leadership is motivated as well. When a new device or piece of equipment was needed, Professor Zyda saw to its immediate procurement. Likewise, in the thesis review process, Professor Zyda's average turnaround time on a chapter was extremely short. David Pratt provided continuous guidance, support and ideas through the dark days of MIDI driver programming in ThinkC, keeping my motivation and spirits up. Dave was always available and willing to help fix a bug or talk through a glitch.

Lastly, I acknowledge the redeeming work of the Lord Jesus Christ in my life, and my personal relationship with God. It has been a difficult two and a half years here, and without the living, loving God of the universe as my constant companion it would have been unbearable. Thank you Lord.

I. INTRODUCTION

A. BACKGROUND

The concept of virtual reality is not new. Virtual reality systems have been in existence in various stages of participant immersion for many years. From the early work with Helmet Mounted Displays (HMD's) of Ivan Sutherland [SUTH 68], to the fictional works *Neuromancer* [GIBS 84] and *Count Zero* [GIBS 86] of William Gibson, and more recently, the "Battletech" game produced by Virtual Worlds Entertainment, virtual reality is rapidly becoming a household concept.

The degree to which a virtual environment succeeds in immersing its user is dependent on the number of the user's senses it can involve and the effectiveness of eliminating external distractions. To this end, many devices such as the HMD are highly successful at blocking out the outside world visually. To more fully immerse the participant, sound cues are vital.

Incorporation of sound cues into graphical simulation and virtual world environments is an area of significant interest in current research and in the literature. Begault and Wenzel have addressed the technical aspects of implementing sound cues into human-machine interfaces [BEGA 90]. Takala and Hahn have focused on modeling sound worlds by associating a characteristic sound or auditory icon with each object in a scene [TAKA 92]. Friedmann et al have done work with synchronization of user motion with rendered graphics and sound output to create a MusicWorld simulated environment [FRIE 92].

This research is an attempt to improve the reality of an existing virtual reality simulator called NPSNET [ZYDA 92] through the inclusion of sound bytes for appropriate events. NPSNET is an ongoing research project within the Department of Computer Science at the Naval Postgraduate School, with the focus of producing a family of low-cost, visual simulators. NPSNET allows the user of the system to explore a 3D virtual world of

terrain databases in a wide scale networked environment. The system is built around several Silicon Graphics IRIS workstations communicating via an Ethernet local area network.

B. OBJECTIVES

The stated objective of this research was to design a flexible, continuous, interruptible, multi-channel sound interface for real time interactive 3D graphics applications. Originally, the intent of this thesis was to incorporate a Prograph™ software application on a Macintosh IIci to fulfill both the interface and sound reproduction roles. As additional funding became available, awareness of interface possibilities grew, and the limitations of Prograph™ become apparent, the sound generation role shifted to a more capable "sound engine", the Emax II 16-bit Digital Sound System by E-mu Systems, Inc. This system allowed the incorporation of MIDI (Musical Instrument Digital Interface) to further enhance the quality, variety, and rapid response of sounds to NPSNET, fulfilling the flexibility objective. Elementary MIDI principles and theory will be discussed in Chapter V.

C. SCOPE

This thesis focuses on the architectural design of the sound system for NPSNET and the supporting software. The issues of networking among the IRIS workstations, as well as the interface between the sound server Indigo Elan and the Emax II sound system are addressed. The individual appendices address the use of various sound conversion programs and utilities, as well as some of the more important procedures used to operate and maintain a working sound library on the Emax II sound system.

D. THESIS ORGANIZATION

Chapter II provides an overview of the individual pieces of hardware used to generate, modify, transfer, compose and play sounds. The discussion includes various equipment configuration schematics to help clarify the software interfaces of later chapters. Chapter III gives a brief coverage of the various application software used in conjunction with the

hardware of Chapter II. In Chapter IV, the implementation of sound as a feature of NPSNET is presented. The interfaces between the various pieces of equipment involved and the inter-workstation networking features incorporated to support the IRIS sound-server form the basis of the Chapter IV. Basic Musical Instrument Digital Interface (MIDI) history, theory, timing, and instruments are the topics of Chapter V. The final chapter includes a brief summary and proposes some future research possibilities for networked sound.

II. HARDWARE OVERVIEW

A. SOUND CREATION, MODIFICATION, SAMPLING AND STORAGE

1. Macintosh IIci and Associated Peripherals

The Macintosh IIci is a versatile, easy to use platform for the collection, modification and storage of sound files. The various sound manipulation software applications provide additional ease in incorporating a wide variety of sounds in a sound library. The Macintosh used in support of the current configuration of NPSNET runs on operating system version 7.0, and is connected to the local area network (LAN) via an Ethernet connection (using the Apple EtherTalk card in one of the NuBus™ expansion slots). A wide variety of attached peripherals give this Macintosh-based sound system extensive capabilities and excellent flexibility.

The Ethernet connection proved valuable in collecting off-site sound files from various FTP (File Transfer Protocol) sound archives. An alternate method involves gathering sound files using a unix account, moving them to the scratch directory on the local virgo server, then transferring them to the Macintosh with the TOPS™ application. This process will be discussed in detail in Appendix A.

Due to the special features incorporated in the Macintosh IIci, it is especially well suited for sound and audio applications. The heart of the Macintosh IIci is a 32-bit Motorola 68030 microprocessor, running at 25.0 MegaHertz (MHz). Additional special purpose floating-point math coprocessor, Motorola 68882 (25.0 MHz) and Sound Accelerator (discussed in paragraph d. below) cards provide even better performance. These enhancements prove invaluable in recording and editing sounds using the software applications discussed in Chapter III, as most of them are CPU intensive.

The following paragraphs give brief descriptions of the different externals and the specific functions they perform within the sound creation, modification, sampling and storage environment in support of NPSNET. The key players in concert with the Macintosh

IIci are connected in a daisy chain fashion via their SCSI ports. SCSI (Small Computer System Interface) is an industry standard hardware and software specification that allows high-speed data transfers between different pieces of equipment [E-MU 89]. The Macintosh sound system daisy chain consists of the Macintosh CPU (SCSI ID- 7), the internal hard disk (SCSI ID- 0) and three external devices: the Quantum 210MB external hard disk (SCSI ID- 1), the Syquest 44MB removable hard disk (SCSI IDs- 4,6), and the Apple CD-ROM (SCSI ID- 3). The order of devices and their SCSI ID's is depicted graphically in Figure 1.

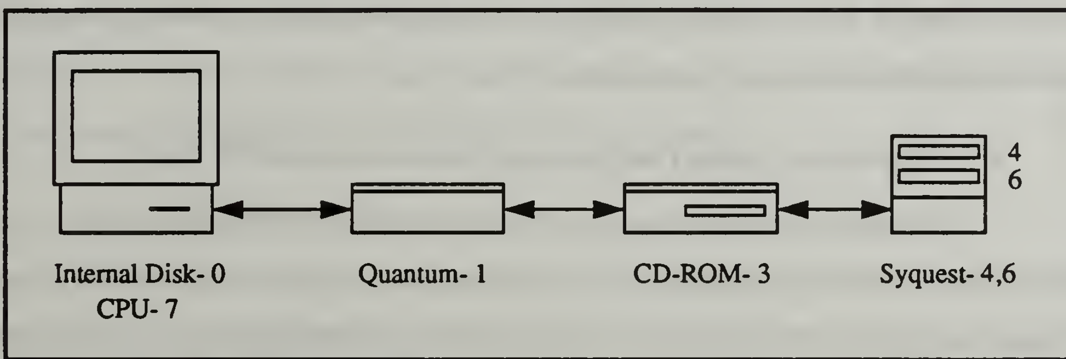

Figure 1 Macintosh SCSI Daisy Chain

a. Quantum 210MB Hard Drive

The Quantum drive, by virtue of its large storage capacity of 210 Megabytes, is the primary software application and sound library repository. The disk is named "Zydaville" in honor of Professor Michael J. Zyda and the small town in NPSNET. There are no partitions on the disk and it was last optimized on August 14, 1992.

b. Syquest 44 MB Removable Hard Drive

Owing to the virtually limitless storage inherent in removable drives, the Syquest disks are used for back up of all system and application files as well as archiving of library sounds. The primary removable disks used with the Syquest drive are "Dulcinea" and "Rocinante," mounted in drive ID's 4 and 6 respectively. Dulcinea holds backup copies of application software and Rocinante contains the backup library of various sounds.

c. Apple CD-ROM

Compact disc sound is renowned for its high fidelity due to its digital nature. For this reason, various sound effects compact discs have been used as the primary source for NPSNET sound recordings. Using the MacRecorder™ application in the stereo recording mode with two digitizers, and a compact disc as the source, excellent quality sound files may be generated. A discussion of MacRecorder™ may be found in Chapter III and the specifics of recording from CD-ROM are located in Appendix B. The device itself is operated by the CD Remote application located in the Apple pull down menu. The controls are similar to any standard compact disc player and the control panel may be operated independently of other applications (using system 7.0).

d. Digidesign Analog Interface and Sound Accelerator™ Card

The initial version of sound generation for NPSNET was based solely on the Macintosh and its sound capabilities. This configuration was discussed briefly in Chapter I. To elaborate slightly, the main IRIS workstation in the NPSNET laboratory, gravy1, sent sound commands as sound file names to the Macintosh IIci via an RS-232 device port. The Prograph™ application FontesTalk II written by a former graduate student, Kevin Fontes, received the filename via the modem port of the Macintosh, searched the system folder for the filename and played the sound. This is a very time consuming process.

The Sound Accelerator digital audio card used in conjunction with the Analog Interface greatly enhances the sound performance characteristics of the Sound Designer II™ application on the Macintosh as well as the original FontesTalk II program. Studio Vision, an application also specifically designed to work in concert with this hardware, is described along with Sound Designer II in Chapter III. The Analog Interface complemented by the Sound Accelerator™ card's playback capabilities provide real time 16-bit compact disc quality stereo sound and recording. Digital recording may be performed at sampling rates up to 44.1 kHz to hard disk, using a Macintosh II or SE/30 [DIGI 90], which is compatible with the sampling rates offered by the Emax II.

B. MUSICAL INSTRUMENT DIGITAL INTERFACE (MIDI)

1. Emax II 16 Bit Digital Sound System

The primary function of the Emax II sampler, described in [E-MU 89], in the NPSNET laboratory is that of digital sound generation and small scale storage. The current configuration of the Emax II sampler has eight megabytes (MB) of RAM, a 40 MB internal hard drive and a 3.5 inch high density capable floppy drive. Future planned expansion includes a 300 MB external hard drive for sampled sound archiving.

In addition to synthesizing sounds, Emax II digitally records, or samples, real world sounds into its memory with 16-bit, CD quality in either mono or stereo. Pre-sampled sounds can be stored on the Emax II's built-in hard drive, on an external hard disk drive, or on double-sided, double-density (DSDD) 3.5 floppy diskettes.

a. Emax II Basics

As a recording device, the Emax II is conceptually similar to a tape recorder, however, the recording method is different. Emax II converts incoming audio signals into numbers by sampling the incoming signal level at a maximum rate of 39,062.5 times per second.

Audio levels are sequentially recorded to memory virtually instantaneously for future use. Samples take up significantly more memory than simple mono voices. For example, at the highest sampling rate, a three second sound would require 3 x 39,062.5 or 117,187.5 samples. It is easy to see how a library of even moderate quality sampled sounds of 5-10 seconds can quickly occupy a large portion of memory. The Emax II also provides sample rates of 20.0 kiloHertz (kHz), 22.050 kHz, 22.778 kHz, and 31.250 kHz, in addition to the maximum rate of 39.0625 kHz.

b. Banks and Presets

The *bank* contains all of the sound memory for the Emax II. This includes preset, voice, sample and sequence data. The bank may be considered as the central

warehouse for all of the Emax II data. This "warehouse" provides temporary (volatile) storage until permanently saved to hard disk or floppy. The hard disk is the preferred method of permanent storage because of greater capacity and faster disk access time. Table 1 gives a brief comparison of access times for saving and loading a 1 megabyte bank to and from the hard disk and floppy drives.

Table 1: EMAX II DRIVE ACCESS TIME COMPARISON

Drive Type	Save 1MB Bank	Load 1MB Bank
Hard Disk	12 seconds	6 seconds
Floppy Drive	120 seconds	50 seconds

A sample is a digital recording of a sound. Samples can be created using Emax II or one of the Macintosh applications discussed in Chapter III. Experience has shown that the number of samples that will fit in a given preset is limited by the amount of RAM available on the Emax II. With eight megabytes of RAM, approximately two minutes of samples may be loaded by any one preset. When more than two minutes worth of samples are loaded, the Emax II sampler begins to perform erratically; playing more than one sample per keystroke, truncating samples, playing high pitched squeals at the end of samples, etcetera.

Raw samples can be digitally processed with Emax II's DSP facilities to create a *voice* [E-MU 89]. While voices are similar to samples, a voice generally refers to a sample which has been processed on the Emax II sampler, and a sample refers to raw digital data or an imported sound file. Individual voices can be saved on disk and loaded from disk as part of a *preset*. Presets store voices/samples in a bank- in other words, a preset is a subdivision of a bank. The bank can hold up to 100 presets, numbered from 0-99. The Emax II is then capable of storing 100 banks (0-99), however, because of the memory

limitations of the hard disk, this may not be realistically achieved without additional external drives.

Sequences are primarily used in conjunction with the musical capabilities of the Emax II. A *sequence* is usually generated by entering the SEQUENCER MANAGE mode selecting an empty sequence, pressing RECORD, then PLAY. The keyboard player then plays a given selection and presses STOP. This procedure is covered in more detail in [E-MU 89] on pages 38-39. An alternate method involves creating a sequence on the Macintosh and downloading it to the Emax II using Supermode. Sequences may prove useful in generating synchronized sound for scripted engagement demonstrations of NPSNET in future applications.

c. Modules and the Sequencer

The Emax II has six main modules and a sequencer module. A module controls a particular aspect of operation of the Emax II. The main modules include: MASTER, SAMPLE, DIGITAL PROCESSING, PRESET MANAGEMENT, PRESET DEFINITION, and DYNAMIC PROCESSING [E-MU 89]. Each module contains individual functions which perform specific actions such as: adjusting internal settings, changing defaults, saving presets, digital and dynamic signal processing, plus a wide variety of others. Some modules have functions nested as deep as three levels. The basic functions are listed under the module they are located in on the face of the Emax II sampler for quick reference. The SEQUENCER module is primarily used to record sequences as discussed briefly in the preceding paragraph.

2. Studio 3 MIDI Interface

Studio 3 provides a standard MIDI interface for the various pieces of equipment that are MIDI capable. The Studio 3 is a MIDI interface incorporating programmable MIDI output selects with a built in SMPTE/MIDI timecode converter. SMPTE Time Code is an international standard, created by the Society of Motion Picture & Television Engineers,

which specifies a format and modulation method for digital code to be recorded on a longitudinal track of video and/or magnetic tape. [OPCO 90a]

SMPTE Time Code was first adopted as a standardized interface protocol in 1969. Chapter V discusses MIDI compatibility with SMPTE Time Code in greater detail as well as the role of MIDI Time Code (MTC).

Currently, Studio 3 is primarily used as a monitoring device when transferring files from the Macintosh via Sound Designer II™. See Appendix C for the details of this procedure. Figure 2 shows the layout of the front panel of Studio 3 for user reference.

In future versions of NPSNET, Studio 3 may be used in conjunction with the new Silicon Graphics VideoLab hardware recently installed in the graphics laboratory to synchronize a prerecorded video track with sound bytes. The key to this arrangement involves recording the video with SMPTE Time Code and synchronizing it with the sounds on audio tape. The SMPTE Time Code signal on the audio tape can then be converted into MIDI Time Code (MTC) with the Studio 3 output going into the Macintosh or some other appropriate recording medium. [OPCO 90a]

Figure 2 Studio 3 (Front Panel)

The Macintosh IIci and the Emax II sampler are the two primary input/output devices connected to Studio 3 in the present system configuration. Appendix E describes the physical connections between these two devices and Studio 3 within the NPSNET sound system. Figure 3 shows the various rear panel input/output ports of the Studio 3 MIDI Interface. Additional MIDI devices may be added to enhance recording and playback capabilities of the Emax II in the future. With Studio 3 as the central synchronization and integration point, future MIDI expansion possibilities are virtually limitless.

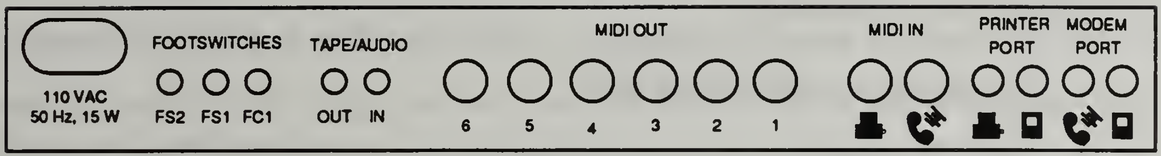

Figure 3 Studio 3 (Rear Panel)

3. Carver Amplifier and Infinity Speaker System

The Infinity speakers and Carver amplifier physically generate the audio of NPSNET. The amplifier and speakers do not actually handle MIDI data, but the Emax II interfaces with the amplifier by means of a special cable. The cable connects the Emax II on one end via a male phono plug, and a pair of RCA plugs provide the stereo input to the amplifier on the other end. The Emax II provides a simple stereo audio signal to the amplifier based on the sampled sound generated by the appropriate “note on” command from the keyboard or NPSNET. The audio level in the laboratory is adequate using this sound amplification system, however, added virtual realism may be obtained by the inclusion of additional amplification and speakers.

4. Apple MIDI Interface

This small device simply receives serial data from an RS-422 protocol, DIN-8 cable, converts the serial data to MIDI protocol, and sends it out via a 5-pin MIDI cable. The converter is also capable of receiving MIDI data and converting it to serial data to reply to the sending device. In the current NPSNET configuration, an IRIS Indigo Elan is used as the serial data sending device and the receiver is the Emax II sound system.

The Apple MIDI Interface can also be used in conjunction with a Macintosh, for which it was originally designed, provided a MIDI driver program is installed. The DIN-8 connector is simply inserted into the Macintosh's printer or modem port and the application software must be told which port was selected. Any MIDI driver program should have the flexibility of testing either port to determine from which port it should send and receive data.

C. NPSNET HARDWARE

1. Sound Server - IRIS Indigo Elan

The IRIS Indigo Elan is a low cost graphics workstation built by Silicon Graphics Incorporated (SGI). The Indigo has been used for this application instead of higher caliber IRIS models primarily because of its device port RS-422 protocol compatibility. One additional advantage of using the IRIS Indigo vice the 4D/240VGX or 4D/120GTX models is that the Indigo runs at an extremely fast 33 MHz (see Table 2, "SILICON GRAPHICS IRIS WORKSTATIONS," on page 15 for comparison figures). The speed coupled with a large 48 megabyte main memory make the Indigo a logical choice for handling the sound server role in NPSNET.

Rapid response time is a key factor in rendering sound bytes. To ensure realism, the time delay between player action and system response (in this case, an appropriate sound effect), must be minimized. The sound byte response time experienced by a networked NPSNET player ranges between 400 and 850 milliseconds (msec) with the average being approximately 670 msec. This performance level is maintainable even with multiple players, generating virtually continuous sound messages and continuous background sounds. The rapid response time is primarily due to the sequential fashion in which the Emax II handles multiple MIDI "note on" commands.

The Emax II assigns mono sound channels on an incremental basis- as a note on command arrives, the next hexadecimal channel number (0-F) is used unless a specific channel number is passed as part of the command. Thus, sixteen channels are nominally available, if Stereo Voice is used however, the Emax II has a 32 channel effective capacity. A few limitations exist when using Stereo Voice: 1) the primary and secondary voices must be assigned to the same keyboard range, 2) Both primary and secondary voices must have the same original key (for musical applications), and 3) Both primary and secondary voices must have the same sample rate. [E-MU 89]

Violation of the above rules after a stereo voice has been created will yield unpredictable and often undesirable results. An additional method of attaining 32 channel capacity involves the use of a second Emax II connected to the MIDI Out port of the primary Emax II. The main Emax II must have MIDI Overflow mode selected to take advantage of this feature [E-MU 89]. The current version of NPSNET runs very well using 16 channels.

A minor problem occurs when MIDI data is sent back to the IRIS Indigo Elan by any of the Module or Sequencer buttons or the Transpose, Drive Select, Load Bank or Enter buttons on the Emax II. When these buttons are pressed while the Output/Thru 5-pin MIDI cable is connected to the MIDI Interface, and a user is logged into the Indigo, the Indigo may close its serial port by removing access permission. This anomaly is intermittent in nature and can be prevented by leaving the Output/Thru 5-pin MIDI cable disconnected. It is not necessary for the Output/Thru cable to be connected for the sound interface between the IRIS Indigo and the Emax II to have a fully functional NPSNET sound system in the present configuration.

To run NPSNET with sound, the sound server IRIS can be used independently or networked to provide sound to the other IRIS workstations. The command line option to use NPSNET with sound from the sound server and any other workstation is given in Figure 4. The user must be in the directory indicated to execute NPSNET. A simple alias to avoid having to remember the lengthy pathname to this directory as well as having to type the entire pathname is shown also. Both the “L” and “l” command line options automatically start NPSNET in the “networking on” mode. This is accomplished in jeep.c by setting the networking flag to TRUE in the switch construct of the main routine (jeep.c is discussed in Chapter IV). See Appendix D for complete system set up procedures.

```
elsie:/n/gravy1/work/dahl
% dem

elsie:/n/gravy1/work2/pratt/simnet/sdis/demo/net
% npsnet L T

gravy1:/n/gravy1/work2/pratt/simnet/sdis/demo/net
% npsnet 1
```

Figure 4 NPSNET Sound Command Line Options

In the event sounds will not play or NPSNET will not run from the commands in Figure 4 on the IRIS sound server, the serial port may have closed. The serial port status can easily be checked by listing permissions on the port. Figure 5 and Figure 6 show examples of good and bad device port listings respectively. The difference between a good and bad device port listing is indicated in the permissions list. To send messages across device port two (/dev/ttyd2) the system must have read (r) permission for that port. Figure 6 indicates that read (r) permission for *group* and *others* is denied. The closed port can be reopened, but it must be done by a user with root or system access. The command to do so is shown in Figure 7.

```
elsie:/n/gravy1/work/dahl
% ls -al /dev/ttyd2
crw-rw-rw- 1 root sys 0, 2 Aug 11 16:49 /dev/ttyd2
```

Figure 5 Good Device Port Listing

```
elsie:/n/gravy1/work/dahl
% ls -al /dev/ttyd2
crw-w--w-- 1 root sys 0, 2 Aug 11 16:55 /dev/ttyd2
```

Figure 6 Bad Device Port Listing

```
elsie:/n/gravy1/work/root
% chmod go+r /dev/ttyd2
```

Figure 7 Open/Enable Device Port

2. Networked SGI Workstations

A wide variety of Silicon Graphics machines are currently in use in the NPS Graphics and Video Laboratory. Table 2 gives a brief summary of the IRIS workstations which compose the local NPSNET network and a brief description of their hardware inventory.

Table 2: SILICON GRAPHICS IRIS WORKSTATIONS

Workstation	Model	CPU's	Clock Speed	Main Memory
gravy1	4D/240VGX	4	25 MHz	64 MB
gravy2	4D/310VGX	1	33 MHz	40 MB
gravy4	4D/120GTX	2	16 MHz	40 MB
gravy5	4D/70GT	1	12 MHz	16 MB
trac1	4D/340VGX	4	33 MHz	32 MB
bessie	Indigo Elan	1	33 MHz	48 MB
elsie	Indigo Elan	1	33 MHz	48 MB
bossie	Indigo Elan	1	33 MHz	16 MB
holmes	PI 30 Elan	1	30 MHz	40 MB

All of these machines are capable of running NPSNET. The machines without a VGX suffix on the model name do not have the ability to perform texturing. It is recommended that these IRISes execute NPSNET with the "T" command line option to turn texturing mode off.

To run NPSNET with sound from a workstation other than the sound server (elsie in the current configuration), the user has two options. The first option requires that the user remotely log in (rlogin) to the sound server and use the command line options from the fourth line of Figure 4 (the "T" option may be omitted if using a texture capable VGX IRIS). This option tends to experience lengthier response times due the greater number of network messages sent back and forth via the LAN. The second option simply requires the user to run NPSNET directly from the sound server as in Figure 4, and from the desired IRIS workstation as in the sixth line of the same figure.

III. COMMERCIAL APPLICATION SOFTWARE

A. STUDIO VISION

Studio Vision as described in [OPCO 90b], is a professional recording tool that combines all of the MIDI sequencing capabilities of Opcode's sequencer, Vision, with the ability to record audio direct to disk in 16-bit linear format at a sample rate of 44.1 kHz. Studio Vision's audio playback quality is equal to the quality of compact disc playback.

One of the main features of Studio Vision software is its ability to integrate MIDI and digital audio recording. Along with fairly extensive sound manipulation features, Studio Vision also includes MIDI event editing within its repertoire. Both analog and digital sound fall within the capabilities of Studio Vision and its hardware companions.

Conversion of analog audio to digital audio (A to D) and digital to analog (A to D) is accomplished by the Digidesign Sound Tools™ (see "Analog Interface", page 6), working in conjunction with Studio Vision. In addition, synchronization of audio to SMPTE timecode can be performed by this hardware/software pair. Chapter II describes the Macintosh hardware that interacts with Studio Vision. Simultaneous recording and audio play back on two separate audio channels is possible with Studio Vision and Sound Tools as well.

Recording with Studio Vision at the maximum quality of 44.1kHz uses 5 megabytes per minute of monophonic digital audio. The audio portions of Studio Vision recordings are stored in Sound Designer II format. Studio Vision is also capable of playing audio files stored in Sound Designer, Audio IFF, and Dyaxis formats as well as Sound Designer II format.

Due to the complex nature of Studio Vision with its plethora of features, and the simpler nature of Sound Designer II and MacRecorder, Studio Vision has not been extensively used in the Macintosh sound creation regime. Future inclusion as an integral

part of MIDI generated sound is recommended as the more powerful aspects of MIDI are incorporated to add further audio realism to NPSNET.

B. SOUND DESIGNER II

Sound Designer II™, discussed in [DIGI 90], is a Macintosh-oriented application that was created to function as the central control for hard disk recording and sound file editing. A prime difference between Sound Designer II and the other applications addressed in this chapter is the ability to perform two different types of editing: destructive and non-destructive.

Destructive editing involves permanently rearranging or modifying the actual sample levels to alter the way a sound file sounds. This feature is particularly useful in preparing sound bytes for transfer to a sampling device for playback. One drawback of this type of editing is that it is RAM intensive and the monitor refresh rate is fairly slow when performing large changes. Conversely, if the intent is to use Sound Designer II purely for its hard disk recording capability, non-destructive editing is the preferred course of action.

Using the Sound Accelerator card (see page 6), all Sound Designer II files can be played in full 16-bit stereo CD quality on the Macintosh. Sound Accelerator is also capable of playing back mono and stereo sound files that are larger than the available main memory (RAM) of the Macintosh. In practice, the only limit to the size of sound files Sound Accelerator can play, is the amount of available hard disk space. An additional feature of Sound Designer II is its ability to synchronize to SMPTE Time Code (see page 9) via MIDI Time Code (MTC). This is an important attribute for audio-video production applications.

Coupled with a MIDI interface (or direct connection via SCSI or RS422 cable), Sound Designer II allows the user to retrieve sounds from any supported sampling device, edit them, save them on the Macintosh, and exchange them with other samplers. Recording using Sound Designer II, however, is a bit of a chore. A number of preliminary settings must be made, the input must be connected to the Sound Tools (Analog Interface), and a

sufficiently large portion of contiguous hard disk space must be available based on sample rate duration (see Table 3 in Appendix B on page 40).

The primary use of Sound Designer II in this research has been to transfer files from the Macintosh to the Emax II. Some modification of sound files is required prior to transfer to ensure format, sample rate, and stereo/mono modes are in sync with the Emax II sampler. Sound samples are sent from the Macintosh to the Emax II via a special high speed serial communication cable.[1] Cable length has been a limitation in this evolution due to the short length of the cable supplied by E-mu Systems with the Emax II sampler. The inconvenience comes from the necessity of locating the back of the Macintosh within less than a foot of the Emax II.

An interim solution to this dilemma involves connecting the Emax II to the Studio 3 Digital MIDI Interface and connecting Studio 3 to the modem or printer port of the Macintosh. Since the Studio 3 has significantly fewer connections, most of which are much longer than the short SCSI cabling of the Macintosh, this has proved to be an acceptable, although not optimum solution. Future laboratory configurations incorporating rack mounts should eliminate this inconvenience.

Sound Designer II is capable of saving recorded files in different formats as well as importing and editing non-Sound Designer II format files. These formats include Audio IFF, Sound Resource, Sound Designer II Mono and Stereo, and SoundEdit™ (MacRecorder aiff format) files. A discussion of file formats and conversion methods is presented in Appendix A. If disk space is a concern, files can be saved in a compressed format (providing a Sound Accelerator card is installed) at ratios of 2:1 and 4:1. However, saving files in the compressed format actually reduces the amount of sample data in the file and thus the audio quality. With the extensive disk space currently available on the

1. The special cable is constructed with a Macintosh modem/printer port compatible, male DIN-8 plug at one end and a DP-9 RS-422 type connector at the other. These cables are extremely difficult to locate.

Macintosh and its peripherals, memory conservation has not been a concern with the present sound byte library. [DIGI 90]

C. MACRECORDER

The MacRecorder Sound System is actually composed of a combined hardware and software package that makes use of the excellent sound capabilities built into the Macintosh family of computers [FARA 90]. The primary use of MacRecorder in this research has been as a digital stereo recording device. Detailed recording procedures can be found in [FARA 90], and discussion of recommended recording procedures specific to this system follow in Appendix B. The software component of MacRecorder is the SoundEdit™ application which provides the ability to record, edit, enhance, play and store sounds in a more intuitive manner than the complex Sound Designer II and Studio Vision environments. Two simple digitizers which function as digital microphones compose the hardware element of the MacRecorder package.

Based on the nature of recording done in this research, MacRecorder has been more than adequate in both recording and editing roles. The digitizers provide a great deal of flexibility in recording sound bytes. The digitizer is a hand-held device composed of a built-in microphone, external microphone jack, line-in jack, input level knob, and DIN-8 plug [FARA 90]. With two digitizer "microphones" capable of recording anything from the human voice to compact disc in stereo, the only limitation to recording with MacRecorder is to be close enough to the desired sound.

Synthesis of unique sound bytes with the waveform editing features from the SoundEdit Effects menu is a simple process. From simple amplification to the Echo, Flanger, and Bender effects, MacRecorder provides the means to easily generate a panoply of sounds. A number of sounds bytes (such as the famous MoofJet), incorporated in previous versions of NPSNET were created by modifying existing sound files using some of the effects mentioned above. Generation of synthetic space-type sounds (laser, photon torpedo, etc.) with MacRecorder for futuristic versions of NPSNET is in progress.

A minor limitation incumbent with MacRecorder is its inability to send samples to, or receive them from the Emax II. Thus, sound bytes recorded using SoundEdit must be converted to Sound Designer II file format to enable their transfer to the Emax II. As with the prior two applications, SoundEdit supports a variety of sound file formats. These formats include SoundEdit format, Instrument format, Audio IFF, and two flavors of sound resources format [FARA 90]. See Appendix A for a more detailed discussion of these formats.

IV. NPSNET - THE INTERFACE

A. THE SOUND SERVER

An IRIS Indigo Elan functions as the sound server in the current laboratory configuration of NPSNET, with various 4D two and three hundred series VGX machines sending the IRIS Indigo sound message packets to play. The primary task of the sound server is to handle the message packets generated by the other machines on the network, as well as its own message packets, and send the appropriate MIDI command to the Emax II sampler.

NPSNET is composed of a number of routines, which perform a variety of functions from networking, to display rendering and updating, to reading of Object Format Files (OFF). The generation of sound encompasses five of the files which make up NPSNET. These files are dogsncats.c, jeep.c, network.c, sound.c, and sound.h. Sound.c and the associated header file, sound.h, are the only files whose sole functions are sound and MIDI oriented, sound is simply an added feature in the remaining files.

1. Sound.c

Sound.c is a MIDI input/output (I/O) file originally written by Robin Schaufler, modified by Dave Gordon and Tom Benoist, and further altered during this research to interface with NPSNET. The original version of this file was written as a test program to demonstrate the IRIS's MIDI I/O capability. Prior to modification, four functions and a main procedure comprised sound.c. Afterward, it was reorganized into seven functions, no main, and the function calls were embedded within the existing NPSNET code.

Soundplay, soundkill, and CloseSound are the three functions added to sound.c. Soundplay simply sends three MIDI commands: "note on", the note or sound to be played, and the attack velocity via the OutByte function (previously defined in sound.c). Figure 8 shows the command sequence for MIDI "note on" from soundplay. The command

sequences are similar for the soundkill and CloseSound routines, the only differences being the number of calls to OutByte and the hexadecimal MIDI commands sent.

```
OutByte((unsigned char) 0x90); /* note on */
OutByte((unsigned char) sound);
OutByte((unsigned char) 0x64); /* attack velocity */
```

Figure 8 MIDI "Note On" Command Sequence

2. Jeep.c, Dogsncats.c and Sound.h

Sound.h is the header file which holds sound related global variables and #defines for NPSNET. The hexadecimal values for the sounds to be used with the simulation environment are defined there. An example of three of these sound byte definitions is given in Figure 9. The hexadecimal values coincide with the numerical assignments given to the individual keys on the Emax II. For example, 0x3c is the C programming language representation of 3C hexadecimal, which corresponds to middle C on the Emax II sampler. It is purely coincidental that 3C hexadecimal is the value assigned to the note C in the third octave, represented by C3. Appendix C contains a complete listing of the hexadecimal values assigned to each key. Two global flags are also defined in sound.h: *soundflag* and *soundserver*. These boolean variables are used to tell NPSNET whether or not to enable sound, and if the user's workstation is the soundserver, respectively.

```
#define SHOT 0x3c
#define EXPLOSION 0x3e
#define GROUND_BURST 0x41
```

Figure 9 Sound.h Hexadecimal Sound Byte Definitions

Dogsncats.c is a file that contains routines that really don't belong anywhere else in NPSNET, as the name implies. Various events that occur within dogsncats.c require sound bytes, such as shooting (SHOT) and explosions (GROUND_BURST). Calls to network.c to play sounds are the only interface that sound has with dogsncats.c.

Jeep.c contains the main function of the NPSNET simulator. Therefore, jeep.c controls the activation of sound via the global sound flags defined in sound.h. The command line options to do so are shown in Chapter II (see Figure 4 on page 14). If networking is on (enabled by either networking or sound command line options), the main procedure calls a routine from network.c entitled getpackets, and activates networked sound (soundserver = TRUE). Jeep.c makes calls to the procedure in network.c which puts sound byte requests on the network as well.

3. Network.c

Network.c performs the client and server networking functions as described in [BACH 86], pages 382-388. The unix system is a complex programming environment, especially with respect to networking. The following discussion leaves out the details of declaring *arenas* and *barriers*, opening *sockets* and passing *process id's*. The primary focus of the explanation is to describe the flow of a sound byte through the network. Figure 10 graphically portrays the functions and routines called, as well as some of the hardware involved. See Appendix E on page 56 for complete system configuration diagrams.

The procedure that is called from dognscats.c and jeep.c to put sound messages on the network is named sendnetsoundmess (send network sound message). This procedure checks for a TRUE soundflag condition, and constructs a network *message* which contains the *soundname* and the necessary header data, if sound is activated. Putmessonnet (put message on network) is called by sendnetsoundmess, and as the name implies, places the sound message on the ethernet network.

While putmessonnet is putting messages on the network, jeep.c also calls getpackets, another procedure located in network.c. The final value in getpackets' argument list is the soundserver flag. If the soundserver is an active participant on NPSNET (networking = TRUE), the SOUNDMESS (Sound Message) case of the main switch construct will call the procedure getsoundmess. Recall that the network message still contains header data as well as the name of the sound byte. Getsoundmess strips the

soundname out of the network message and passes it to the soundplay function previously described in sound.c, which sends the hexadecimal value to the Emax II to play.

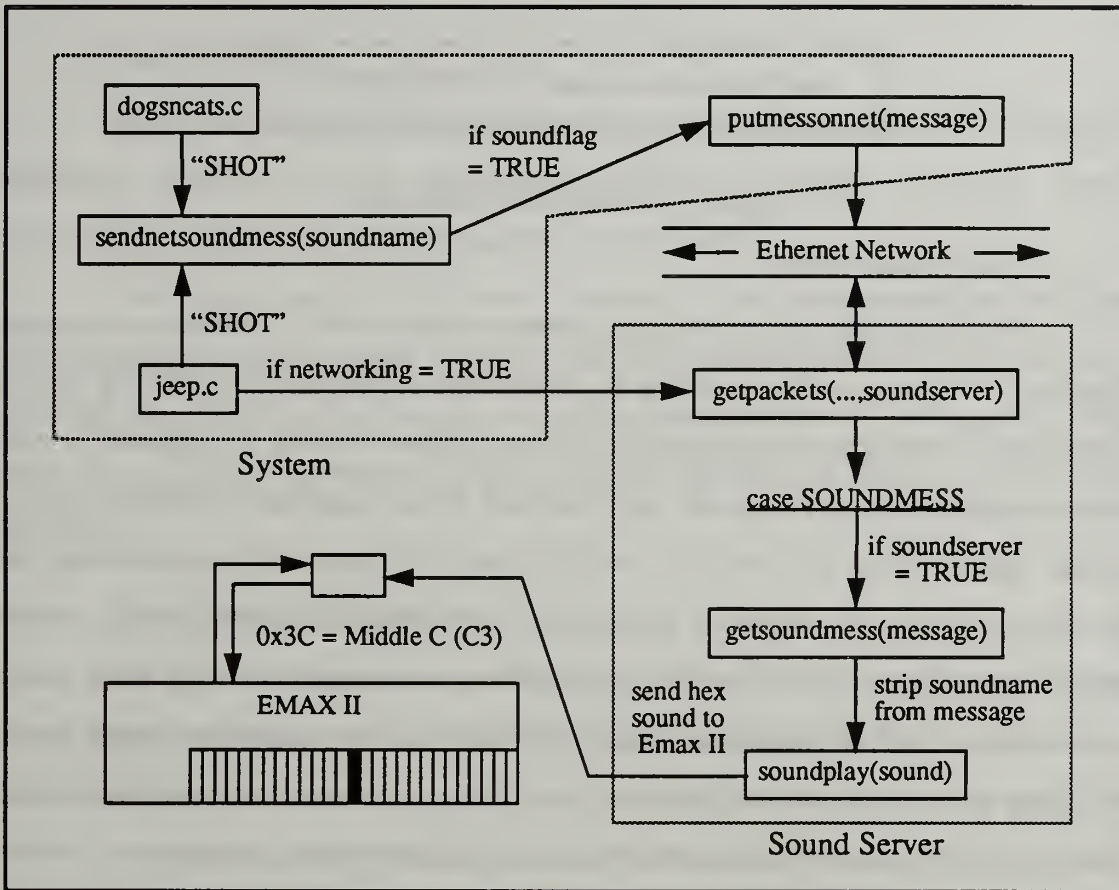

Figure 10 Networked Sound Logical Flow

B. THE IRIS INDIGO ELAN - EMAX II INTERFACE

As described in Chapter II, the hardware interface between the sound server and the Emax II is based on the Apple MIDI Interface (see page 11). The IRIS Indigo has a variety of external device ports, but only three DIN-8 ports, of which only are two capable of RS-422 protocol. Device port one is occupied by the IRIS spaceball in a normal operating configuration, which leaves device port two open for the MIDI Interface. The specific device port to be used must be declared in sound.c as indicated in Figure 11. The code for

sound implementation on NPSNET is fully portable with the exception of this one machine dependency.

```
char *MidiPortName = "/dev/ttyd2";
```

Figure 11 Device Port Declaration

1. Continuous Sound

The implementation of continuous sound has proven to be a difficult undertaking, primarily due to the interface between the sound server and the Emax II. The looping feature in the Digital Processing mode of the Emax II allows the user to "program" a given sound to play continuously when the key is pressed. When a note-on command is sent to a "looped" sound byte on the Emax II, it will play until enough sound bytes have been sent that the looped sound's channel is required by another sound. Attempts to incorporate continuous sound by use of looping or iterative constructs in sound.c have proven unsuccessful as well. Modifications to sound.c have resulted in infinitely looped sounds, the playing of one sound to the exclusion of all others, or momentary continuous sound broken by the next sound byte to arrive. This is an area of ongoing research.

2. Multi-Channel Sound

The original difficulty of implementing multiple channels of sound with the Fontes Talk II Prograph application has been solved through use of the Emax II. The method by which the Emax II performs sound channel assignment is discussed in more detail in Chapter II (see page 12). This assignment process allows the Emax II to handle a large number of sound effects in rapid succession with little or no degradation in response time providing the sound bytes are of a reasonable length (less than five seconds each). The average length of an NPSNET sound is two to three seconds.

V. MUSICAL INSTRUMENT DIGITAL INTERFACE (MIDI)

A. BASIC MIDI THEORY

1. MIDI History

MIDI, an acronym for Musical Instrument Digital Interface, is a communications protocol, a standard way of exchanging information between electronic musical instruments, and between computers and those instruments.

The original goal of MIDI was to provide a common standardized electronic instrument protocol within which these instruments could communicate. The result of the efforts of many in the music industry as well as those in academia in the early 1980's produced the Musical Instrument Digital Interface (MIDI Specification 1.0) in 1983. [HUBE 91]

2. Samplers and Synthesizers

MIDI instruments come in a wide variety of flavors, with a multitude of features as well. Early synthesizers were monophonic, meaning they can play only one voice at a time. Modern synthesizers, such as the Emax II are capable of multiple voice production. Another mandatory feature for a virtual world sound engine is that it be multitimbral (able to produce several voices simultaneously). Samplers are similar to synthesizers, (the terms are often used interchangeably), but samplers have the additional ability of being able to digitally record and playback sound. The Emax II fulfills this role as well. Sampler quality is measured by bit resolution, which is the number of bits used to describe each sample. The dynamic range of 8-bit resolution is divided into 256 levels, while the dynamic range of 16-bit resolution has 65,536 levels, clearly the higher bit resolution produces a much higher quality sound. [YELT 89]

3. Electrical and Hardware Specification

The MIDI data transfer rate, or baud rate, is 31.25 Kbaud (+/- 1%). The transmission is via an asynchronous serial interface with eight data bits- one start bit and

one stop bit for 320 microseconds per serial byte. MIDI data is transmitted or received via MIDI *In* and *Out* ports. The data that passes through these ports are unidirectional, in that the information moves in a one-way fashion. MIDI data always flows out of the Out port and into the In port. A third port type is called the MIDI *Thru* port. The Thru port is primarily used for daisy-chaining of other MIDI devices and simply passes MIDI data received from the In port directly to the device on the other end of the Thru cable. MIDI data passing through a device in this manner remains unchanged and the output is virtually instantaneous [YELT 89].

Anderton [ANDE 86], provides an in-depth discussion of MIDI hardware and theory, from the Voltage Control Oscillator which functions as a tone generator to the Universal Asynchronous Receiver-Transmitter (UART) which is specifically designed to transmit and receive MIDI formatted messages. Anderton's Appendix A provides a complete description of the MIDI 1.0 specification.

4. Channels, Modes, and Messages

The MIDI specification requires 16 channels for receiving and transmitting data. Channels are the primary flow path between instruments for MIDI messages and data. Instruments can be "told" to receive and act upon data on just one channel and to ignore the remaining data they receive on other channels [YELT 89]. Conversely, multitimbral devices can receive input on several different channels at once, playing the appropriate note or sound as well. The speed and flexibility of multitimbral MIDI devices make them a logical choice for the role of the sound engine in NPSNET.

MIDI instruments are capable of operating in one of four different modes:

- Omni On/Poly
- Omni On/Mono
- Omni Off/Poly
- Omni Off/Mono

Omni On/Off refers to how a MIDI instrument will respond to or transmit on the different MIDI channels. Omni mode On means that a device will respond to all channels and is not specifically set to any one channel. Omni mode Off indicates that the receiving device is looking for input on one specific channel. Poly or Mono describes how many voices a device can play. In the Poly modes multiple voices can be played, while in the Mono modes only a single voice can play at one time.

Five different message types exist to support the various features of MIDI instruments:

- Channel Voice messages
- Channel Mode messages
- System Common (All channels) messages
- System Real-Time messages
- System Exclusive messages

Channel Voice messages transmit real-time performance data within a MIDI system. Some examples of channel voice messages are: Note-on, Note-off, Control change, and Pitch bend. Channel Mode messages are all special cases of the channel voice control change message, which affects a given channel's mode of operation. Examples of channel mode messages include Reset All Controllers, Local Control, All Notes Off, and Omni Mode Off/On.

System Common or All Channels messages are transmitted to every device or instrument in the MIDI daisy chain. The reason for this is that no channel information is included in the byte structure of a system message. Huber divides the All Channels messages into three types: System Common, System Real-Time, and System Exclusive. The names of system common messages are indicative of their global nature: Song Position Pointer, Song Select, and transmission of MIDI Time Code (MTC). System Real-Time messages start and stop timing-sensitive devices and are primarily concerned with the synchronization of MIDI devices within the system. Start, Stop, Continue, and System Reset are a few examples of system real-time messages. Customization of MIDI messages

is accomplished with System Exclusive messages. This message type is specific to a unique make and model of instrument, and is encoded with a manufacturer's MIDI identification number [YELT 89]. MIDI Programmers can create tailor made MIDI messages to communicate device specific data of unrestricted length between studio components. [ANDE 86], [HUBE 91]

B. SYNCHRONIZATION AND TIMING

1. MIDI Time Code

MIDI Time Code (MTC) is basically a method of transmitting SMPTE Time Code (see page 9) across MIDI communication channels. MTC uses a format based on location in real time as opposed to a starting position on a track. The basic timing unit is the MTC Quarter Frame message, which is sent 120 times per second, giving a ten-fold increase in precision over MIDI clocking pulses for added accuracy in event synchronization [YELT 89]. MIDI Time Code was incorporated in the official MIDI Specification in March, 1987 [DIGI 90].

2. Compatibility with SMPTE

Transmission of actual SMPTE Time Code over MIDI is not practical due to the size of each SMPTE message. Each frame of SMPTE Time Code is composed of 80 bits of digital information. MIDI's limited bandwidth would rapidly be consumed by the transfer of this quantity of information at a rate of 30 times per second (standard frame rate). In the digital interface context, bandwidth refers to the maximum information transmission speed. [DIGI 90]

The bandwidth of SMPTE Time Code at 30 frames per second with 80 bits per frame is 2.4 Kbaud. To transfer SMPTE Time Code over MIDI a good deal of supplemental data must be included. This additional data overhead would probably interrupt or interfere with the normal operation of MIDI. The transfer of full SMPTE Time Code over MIDI often results in a condition called MIDI Delay. Thus, MTC was developed to improve MIDI compatibility with the preexisting SMPTE standards. [DIGI 90]

VI. CONCLUSIONS AND RECOMMENDATIONS

A. SUMMARY

The primary focus of this research has been the integration and interface of a variety of software applications and hardware systems to provide an enhanced acoustic environment for NPSNET users. Incorporation of additional code segments within various portions of NPSNET files provides the catalyst which draws the IRIS graphics entity together with the Emax II generation of sound. The motivation for the addition of this feature to NPSNET has been to improve the virtuality of simulation by drawing another of the user's senses into the realm he or she is experiencing. Incorporation of aural cues in the virtual world environment of NPSNET has accomplished this goal, as indicated by favorable user reaction to this feature.

B. FUTURE RESEARCH

1. Three Dimensional (3D) Sound

Research is currently in progress by Elizabeth Wenzel at the NASA AMES Research Center on three-dimensional (3D) sound. The focus of Wenzel's work is based on a device called the Convolvotron which is used to perform sound localization in a Virtual Acoustic Displays [WENZ 92]. Wenzel teamed up with Scott Fisher in a research effort to perform real-time digital synthesis of Virtual Acoustic Environments [WENZ 90].

Brenda Laurel has joined Scott Fisher as well in efforts to accurately implement 3D binaural sound. The Laurel-Fisher Team has created a virtual acoustic environment in which the user wears stereo headphones to give the illusion of 3D sound. As the user flies a virtual radio controlled gas powered model airplane within a large virtual room, 4 sound generation sources and 6 reflective surfaces (walls) create the effect of reflected and direct 3D acoustics in the stereo headphones [LAUR 91].

These research efforts are indicative of the solid groundwork that has been laid for three-dimensional acoustic environment research. The added realism 3D sound

contributes to virtual worlds is probably second only to visual cues in determining the quality of the user's immersion in the virtual environment. Rheingold sums it up well: "Humans have two ears; we can swivel them by moving our head, and the differences in the signals detected from those auditory sensors play a key role in our ability to locate sounds in space." [RHEI 91]

Given adequate resources, future aural possibilities for NPSNET are great. For example, inclusion of doppler for approaching or retreating forces may be implemented by varying the pitch of a continuous background tank engine, helicopter, or jet sound byte using specific MIDI commands to a bank of three samplers or synthesizers. Each of these three devices could be used to represent the directional coefficient of a sound in the x, y, and z axes. To provide the necessary spatial representation of sound, the x axis sampler's speakers are oriented in the laboratory to the left and right of the user, the y axis components are located above and (if possible) below the user (or at the user's feet), and the z axis components are placed in front of and behind the user. Figure 12 provides a possible layout for this proposed arrangement, indicating speaker placement relative to the user's position.

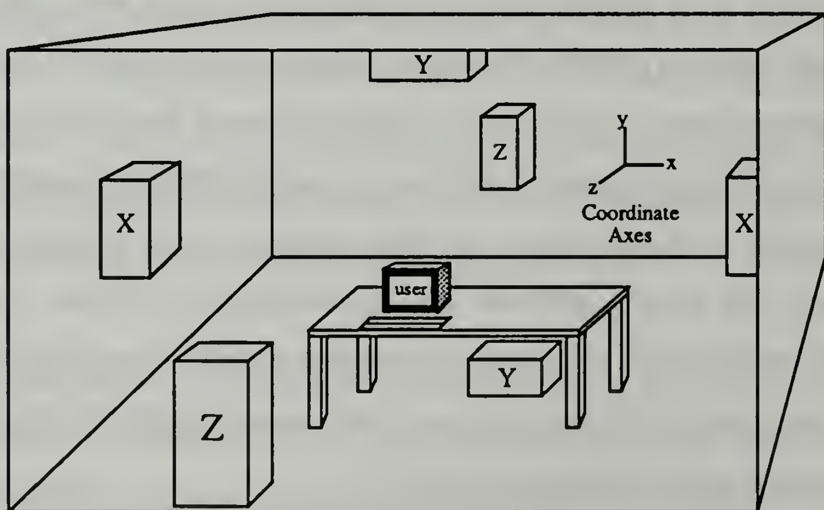

Figure 12 Proposed NPSNET Future Sound Configuration

Physically, this is a relatively simple system to construct, the major constraint being the cost of the samplers or synthesizers and their associated amplifiers and speakers. An additional limitation lies in the number of participants that may be immersed in the acoustic environment using this system. To fully experience the effects of 3D sound, the user must be relatively near the center of the coordinate axes of the speaker system. Thus, six or less players would be the limit, given the current laboratory size.

The difficult aspects of this arrangement lie in establishing the interface with NPSNET and the samplers, as well as embedding the code within NPSNET to accurately compute the MIDI pitch and doppler levels corresponding to vehicular position and direction of travel. Spawning an individual process for each coordinate axis sampler-speaker pair is a possible method of solving the interface dilemma.

2. Canned Speech

Canned speech as it relates to this research involves storing a small library of pre-recorded words or phrases on a bank in the Emax II and playing them in an appropriate sequence to convey a specific message. Whether this involves a robot advertising that it is about to have a collision, or an NPSNET user receiving voice communications from his "commander", canned speech provides additional realism to whatever application it is applied.

Research is ongoing at the Naval Research Laboratory (NRL), Voice Systems Section in Washington, D.C. in the canned speech arena [KANG 92]. Kang and Heide have investigated the feasibility of encoding the human voice in tactical two-way voice communication as opposed to digitized voice generation. According to the results of their work, listeners greatly preferred canned speech over synthetic speech for its higher intelligibility as well as the more natural sound.

Part of the continuing NPSNET research involves hypertext cues for the user at various fixed points on the terrain. The addition of sound bytes (in the media sense) attached to these hypertext cues would provide the user with an even greater ability to

interface with the simulation as well as increasing the rate at which the user learns. The cues may be voice message, or environmental or background effects. In the heat of battle (or simulation) it is often easier to receive information aurally than to divert one's attention to a written message or graphic display.

3. Sonar Acoustic Simulation

The Autonomous Underwater Vehicle (AUV) is another research project that can benefit from the aural interface provided between the IRIS graphics workstations and the Emax II sampler, established in this work. LCDR Donald Brutzman continues to be a driving force in AUV research at the Naval Postgraduate School following completion of his master's thesis and transition from student to faculty. One of Brutzman's potential future research suggestions involves the incorporation of sonar visualization as a feature of the NPS AUV Integrated Simulator [BRUT 92]. Brutzman is also investigating the implementation of canned speech for robot and semi-autonomous forces monitoring and cues in conjunction with the ongoing AUV work.

The principle of sonar visualization allows the user to see and hear the components of underwater acoustics such as, frequency, pitch, and doppler at the same time. AUV missions can be recorded live and played back on an IRIS workstation using a prerecorded bank of standard frequencies on the Emax II, modified as necessary with MIDI doppler and pitch commands to simulate the actual acoustic environment. Additionally, simulations can be performed on an IRIS workstation using the AUV simulator with Emax II audio cues prior to deploying the AUV to provide operators with realistic training and acoustic experience.

C. CONCLUSIONS

The interface work done in this research has far reaching ramifications. By using the excellent sound capabilities of MIDI in conjunction with the superb graphics of the IRIS workstation, a virtual world is just a step away. A few supplemental applications for research have been proposed here, as well as some recommendations for system expansion

and future growth. In the realm of virtual reality, the only limitation is the ingenuity and creativity in the mind of the system designer.

"The door to cyberspace is open, and I believe that poetically and scientifically minded architects can and will step through it in significant numbers." [BENE 92]

APPENDIX A. SOUND FILES

There are two primary methods of collecting sounds to be used in NPSNET or other graphical sound applications: 1) Searching the directories of various sound file archives on the Internet, Milnet, etc.; and 2) Recording sounds from cassette tape, compact disc, or live, all using the MacRecorder application. This appendix addresses the former. See Appendix B, Recording with MacRecorder on page 40 for discussion of the latter.

There is an immense wealth of sound data available to the diligent network sound sleuth. However, there are a few setbacks to this method of sound file acquisition. The primary inconvenience lies in the multiple transfers required for a file to ultimately arrive at the Macintosh, where they can be converted to a recognizable and usable format. The steps involved in this procedure are discussed following the sound file format summary.

A. SOUND FILE FORMATS

A brief summary of some of the sound file formats encountered in this research follows (the "Sound" in "Sound.xxx" refers to any generic sound file name):

Format	Description
Sound.aifc	Similar to AIFF for C programming environments.
Sound.aiff	AIFF stands for Audio Interchange File Format. This file format includes only a data fork, however, if an AIFF file is created or modified by Sound Edit, the information normally stored in the resource fork of a SoundEdit file is stored in an application specific chunk with SoundEdit's signature. See [FARA 90] page 69.
Instrument	A format used by many Macintosh music applications, such as Jam Session and Studio Session. If the file was created or edited by SoundEdit, it will have the same resource fork data as a SoundEdit file. [FARA 90]
Sound.au	AU stands for AUdio files, this format is primarily found in Sun workstation applications.
Sound.bin	BIN represents binary file format. This format often relates to files that are both sound and non-sound format.
Sound Designer	A 16-bit mono format used by the original Sound Designer application. See [DIGI 90] page C-14.

Sound Designer II	An enhanced, 16-bit multi-channel format used by the Sound Designer II application. [DIGI 90]
SoundEdit	The file format used by many sound applications, and is compatible with SoundCap and SoundWave formats. The data fork of a SoundEdit file contains the sound data, and the resource fork contains loopback, selection location, label, pitch setting and various other information germane to the file. [FARA 90]
Sound.hqx	The hqx suffix is generally appended to sound files compressed by a Macintosh compression application program such as Compaq Pro or Stuffit.
Sound.next	A format specifically designed for NeXT computer architectures, also compatible with Sun machines.
Sound resource	Also referred to as 'snd' or 'rsrc' files. Standard 8-bit Macintosh sound formats used by (and located inside of) System software. [DIGI 90] Apple defines two types: Format 1 and Format 2. Format 2 snd files are used by Hypercard, all other file types use Format 1. [FARA 90]
Sound.wave	MS RIFF WAVE format.
Sound.zip	The zip suffix is generally appended to DOS sound files compressed by applications such as PKZIP. Since the NPSNET sound system is a Macintosh based system, no zip files were used due to the lack of conversion software.

B. LOCATION AND TRANSFER OF SOUND FILES

Some wealthy FTP (File Transfer Protocol) sound file sources include:

- San Diego State University (sciences.sdsu.edu, see /pub/sounds directory)
- Stanford University (sumex-aim.stanford.edu, see /info-mac/sound directory)
- University of California, San Francisco (ccb.ucsf.edu see, /Pub/Sound_list directory)
- U.S. Army Information Systems Command, White Sands Missile Range (wsmr-simtel20.army.mil, see file SIMTEL20-MACINTOSH.INFO.8)

Each of these sites may be accessed by typing "**ftp sitename**". When asked for name or userid, enter **anonymous**. Observe the login instructions regarding password entry, and follow the pathname to the directory for the desired site. Once a desirable file is located, **get** or **mget** the file and transfer it to the /scratch directory on the virgo server.

Files stored within the hierarchy of the /scratch directory are accessible by the TOPS™ application on the Macintosh IIci located in the GRaphics And Video LaboratorY (GRAVY) . Using TOPS, open an appropriate folder on the Macintosh for placing the new sound file, as well as opening the file in the /scratch directory of the virgo server. Once the destination folder and source folder are open and the desired file is highlighted, select the **copy** option to transfer the sound file from unix to the Macintosh. The final step may require use of appropriate conversion or decompression software to prepare the sound file for transfer to the Emax II.

C. CONVERSION OF SOUND FILES USING SOUNDHACK

1. SoundHack Background

SoundHack v0.60 is a Macintosh soundfile manipulation application written by Tom Erbe at the Center for Contemporary Music, Mills College, Oakland, Ca. It is capable of converting virtually any file into a variety of soundfile formats. It can also perform soundfile convolution, phase vocoding, binaural filtering, amplitude analysis, and gain change.

SoundHack can read and write the following formats: Sound Designer II™, Audio IFF, IRCAM, DSP Designer and NeXT .snd (or Sun .au). It can read (but not write) raw data files, and can read and write 8-bit uLaw, 8-bit linear, 32-bit floating point and 16-bit linear data encoding.

2. Sound File Conversion

1) Start the SoundHack v0.60 application, it will come up with a file selection menu. Select the "**Cancel**" option.

2) Select the "**Open Any...**" option from the File menu. This will allow the user to open any file, not just those recognized by SoundHack as sound files.

3) Locate the desired file and open it. If an error occurs, simply click "**OK**" and when file the opens, select "**Header Change...**" from the Hack menu. Set Channels: to "**1**", Format: to "**16-Bit Linear**" and **Save Info**.

4) To save the modified file (in a usable format), select "**Save a Copy**" from the File menu.

5) In the Output Soundfile Format window, select "**Sound Designer II™**" and ensure **16 Bit** is selected also. Click on OK.

6) Choose an appropriate directory on the Macintosh to store the new Sound Designer II™ file and Save. For consistency and organizational purposes, using the default ".sd2" (Sound Designer II) suffix is recommended.

APPENDIX B. RECORDING WITH MACRECORDER

A. OVERVIEW

The MacRecorder Sound System as used in support of NPSNET consists of two MacRecorder digitizers (hand-held sound input devices), and SoundEdit™ (a sound editing, playing, and storing application). Additional HyperCard features are included with MacRecorder but were not used in this research.

When recording sounds in any environment, an important consideration is memory usage. The ever present trade off between required memory and sample quality, is illustrated by the following table from the MacRecorder Sound System User's Guide. Table 3 provides some good guidelines for estimating memory requirements for sound storage based on recording sample frequency [FARA 90].

Table 3: SOUND SAMPLE RATE VS. REQUIRED STORAGE

Sampling rate or compression ratio	Bytes needed to store one second of sound	Seconds of sound stored per 1 MB of disk space
44 KHz	44 Kbytes	22.5 seconds
22 KHz	22 Kbytes	45 seconds
11 KHz	11 Kbytes	90 seconds
7 KHz	7 Kbytes	135 seconds
5 KHz	5 Kbytes	180 seconds
3:1	7 Kbytes	135 seconds
4:1	5.5 Kbytes	180 seconds
6:1	3.67 Kbytes	270 seconds
8:1	2.75 Kbytes	360 seconds

MacRecorder may be used to record virtually any sound that can be played within range of the digitizers. With two digitizers, stereo sound can be recorded on two tracks, with each track independently editable. Stereo sound may be recorded with only one digitizer, however, it is a difficult process to accurately synchronize the left and right channels. Recording stereo sound with only one digitizer is discussed at the end of the normal recording procedures.

B. RECORDING SETUP

1) Determine the source from which the recording will be made. MacRecorder is capable of recording from virtually any device that has a mini-plug connector, or can be converted to mini-plug input. The three machine types that are envisioned as primary input devices are the CD-ROM discussed in Chapter II, cassette tape player, video cassette recorder (VCR), and of course, the human voice.

2) Set up the sound system and appropriate input source as described in Appendix E. Ensure the CD-ROM is turned on before the Macintosh or the Macintosh must be re-booted to initialize the CD Remote application. Connect the source device to the MacRecorder digitizer via the mini-plug line input.

3) Start up the Macintosh.

4) Once the boot is complete, select **Chooser** from the **Apple** menu, and make AppleTalk **Inactive**.

5) Start the SoundEdit application by double clicking on any SoundEdit sound file or by opening the **MacRecorder** folder, which is in the **Sound** folder on the **Zydaville** drive. Inside the MacRecorder folder to the upper left is the **SoundEdit** icon, double click to start the application.

6) Select **Recording Options...** from the Settings menu, and make the following settings:

Recording Type	**22KHz** For the best quality sound. Use this sample rate especially for compact disc recording. Consult Table 3 above to determine memory requirements based on estimated sound file duration.

Mode	**Mono** If a single digitizer is used. **Stereo** If two digitizers are used, or if left and right channels are to be recorded separately using only one digitizer.
Connection	In the Mono mode, the user must declare whether the digitizer is connected to the modem port or printer port. Select the **Modem Icon.**
Left	In the Stereo mode, the user must declare whether the **Left** channel digitizer is connected to the modem port or the printer port. Select the **Modem Icon.**

Click on **OK** to make the settings and continue, **Cancel** to abort.

7) Select **User Options** from the **Settings** menu and set to the loudest setting (SoundEdit files have notoriously low volume levels).

8) If recording from compact disc, insert the desired CD in the special CD caddy and load the caddy in the CD-ROM player. Select **CD Remote** from the **Apple** menu and using the controls displayed, sequence to the position on the track to be recorded. Select **Play**, reverse **Scan** 5-10 seconds, and **Pause**.

Note: Time is displayed in two modes in the CD Remote application, time remaining and elapsed time. The **time remaining** mode allows the user to start the CD prior to the anticipated track position, re-enter SoundEdit and start the recording when the time display counts down to the desired location. When SoundEdit begins recording the time display will freeze, and the user must remember roughly how long the track is to know when to stop recording.

9) If recording from cassette tape or video tape, cue the tape to approximately 5 seconds prior to the desired sound.

10) Select **New** from the **File** menu to open a new file in which to record, if a new "untitled" file is not already open.

C. NORMAL RECORDING

1) Start the sound source:

- If using the audio microphone, position it near the source, and rehearse the word(s) if speech is to be recorded.
- Click on the CD Remote control panel to bring the application to the front on the desktop. Click on pause, then immediately click anywhere on the SoundEdit window and position the cursor over the microphone icon to the far left of the window.

2) Just prior to the beginning of the sound segment, click on the microphone icon and release, leaving the cursor over the icon, keeping the mouse <u>motionless</u>. To stop recording, move the mouse or click on the microphone icon again.

<u>Notes on setting levels:</u>
One difficult aspect of recording with MacRecorder is setting the input level to obtain good quality recordings. There are three levels that must be set:

- The level of the original recording (CD, cassette tape, video tape, voice)
- The output level set on the device (CD, cassette player, etcetera)
- The recording level set on the MacRecorder digitizer

Using unamplified input from the CD-ROM, set the level of CD Remote and the digitizer to maximum and set the CD-ROM level to maximum, then back it off one third to one half turn.

The ideal recording waveform will fill the display window from top to bottom with no peaks extending beyond these limits. If the waveform is too small (in amplitude), the sound will be too soft. If the waveform amplitude is too great, the sound will be clipped and produce distorted sound.

For other devices, test the level by recording a short patch of sound repeatedly until the waveform fills the display window with no distortion.

For voice, set the digitizer's recording level to the middle of its range and adjust for the individual user. This level may be further adjusted by varying the distance from the microphone to the speaker's mouth. Optimum distance for normal voice recording is approximately 3-5 inches from mouth to microphone.

3) If the recording is not satisfactory, simply double click on the entire waveform, delete it using the **delete** key or the **Cut** option from the **Edit** menu, and re-record the sound (steps 1 and 2).

4) Once an acceptable level has been achieved **Cut** the unnecessary "lead in" and "fade out" portions of the sound to minimize memory usage and "clean up" the sound.

5) Save the sound file using the **Save** option from the **File** menu. Select **Audio IFF** format and append **.aiff** to the filename in the **Save as:** block. Use a descriptive name and save the file to an appropriate location (**SoundEdit Sounds** folder) by clicking on **Save**.

<u>Note</u>: It is a good policy to make a back up of the sound file and store it with the back up sound files on one of the Syquest removable drives (Rocinante), especially if the file was difficult to record or required a great deal of editing.

6) If the file is to be sent to the Emax II sampler, select **Quit** from the **File** menu and exit SoundEdit. Start Sound Designer II as described in paragraph C of Appendix C (page 49). To open the sound file just created, select **Open** from the **File** menu. When the file selection menu appears, ensure the **Audio IFF** box is checked, or the sound filename may not appear. Proceed with step 4 of Appendix C.

D. RECORDING IN STEREO WITH ONE DIGITIZER

Recording in stereo with a single digitizer is performed in basically the same manner as recording a single channel in mono. A new key stroke-mouse click combination must be learned prior to undertaking this procedure. The **Option-Click** involves holding down the **Option** key on the Macintosh keyboard while clicking the mouse in a specific region of the display. This special action is done to shift back and forth between the left and right channels in the window while recording and editing the channels individually.

1) Perform the recording setup as described in paragraph B, steps 1-6. In step 6, ensure that **Stereo** is selected as the **Mode** option, and that the modem port is selected for the left channel by clicking on the **Modem Icon** beneath it.

2) When recording in stereo with a single digitizer, the left channel should always be recorded first. Select an insertion point for the left channel by using the **Option-Click** in the upper half of the display.

3) Record the left half of the stereo track as described in steps 1-3 of paragraph C above, remembering that any modifications or deletions to the left channel alone require an **Option-Click** on that channel. If a normal mouse click is performed, both left and right channels will be selected.

4) Once the left channel has been satisfactorily recorded, select the left channel by using a double **Option-Click** on the upper half of the display.

5) Select **Cut** from the **Edit** menu, and cut the upper waveform.

6) **Option-Click** in the lower half of the display to select the right channel. Select **Paste** from the **Edit** menu to paste the sound into the right channel.

7) Perform steps 1 and 2 (of this procedure) to record the left channel again.

8) Perform steps 4 and 5 from paragraph C to clean up the sound and save the file to an appropriate location.

For a more in depth discussion of these procedures consult [FARA 90], pages 127-129.

APPENDIX C. EMAX II SAMPLER PROCEDURES

The Emax II Sampler by E-mu Systems is an extremely versatile sound generation and modification device. However, as with most technical support documents, the user's manual is somewhat unintuitive in many areas. For this reason, various vital procedures are presented here in greater detail.

Procedures:

A. Saving a Bank to floppy disk(s)
B. Creating a new Bank
C. Transferring Files to the Emax II from the Macintosh using Sound Designer II™
D. Hexadecimal Values for Emax II Keys
E. Current Emax II Bank Presets

The following conventions will be observed in the ensuing procedures.

1) When a key from the SEQUENCER or MODULES section of the sampler is to be pressed it will be written in **bold print** as it appears on the face of the sampler. Additionally, names in these procedures will be written as they appear on the sampler (e.g. UPPER CASE or lower case), to aid in recognition during execution.

2) When the **ENTER** key is to be pressed, it will generally follow a display of the sampler data window, and indicate [flashing] if the LED below it is flashing.

3) Displays of the sampler data window will be indented and set apart from the text to avoid confusion.

4) Textual equivalents of numeric key options will follow the number in brackets to help the user follow the command sequence more easily.

5) The characters "xx" following a B (bank) or P (preset) are used to indicate the desired Bank, Preset number or other indicated numeric entry (floppies, etc.).

A. SAVING A BANK TO FLOPPY

1) Select **MASTER**, then option 8 [SelectedToFlpy].

Backup SCSI 1
Need xxx Floppys **ENTER** [flashing]

2) At the following prompt select the low Bank to be saved using the DATA slider:

Select Low Bank
Bxx LowBankName **ENTER** [flashing]

3) Similarly, select the high Bank to be saved using the DATA slider (if the user desires to save only one Bank, then the High Bank and Low Bank will be the same):

Select High Bank	
Bxx HiBankName	**ENTER** [flashing]

4) The sampler will then ask if you wish to perform the save:

Save to xx disks?
Bxx LowBankName

The user must then select **ON /\ YES** to save or **OFF \/ NO** to skip to the next Bank.

5) When the **ON /\ YES** option is selected, the user will be prompted to insert disks as necessary:

Backup in Progress	
Reading SCSI 1	[No action required.]

6) Insert the first disk (it must be formatted beforehand using **MASTER 5**), and then press **ENTER.**

Takes xx disks	
Insert disk 1	**ENTER** [flashing]

B. CREATING A NEW BANK

1) Insert a blank Emax II formatted disk. Note: this procedure may be accomplished from the hard disk, however, it is not recommended due to possible modification of existing Banks/Presets.

2) Select the floppy drive: press **DRIVE SELECT**, then enter **0** [SCSI 0: Floppy].

3) Create a new *Preset* in an empty Bank on the floppy: select **PRESET MANAGEMENT**, option **3**.

PresetManagement	->Create Preset xx
[1-8] / Slider	->Select A Preset

Use the DATA slider or numeric keys to select the Preset number, xx, then press **ENTER** [flashing].

4) The display window should now say:

Pxx Untitled

5) Rename the Preset to the name desired for the new Bank: select **PRESET MANAGEMENT**, option **6**.

Rename Preset xx
Select A Preset **ENTER** [flashing]

The display window will show the default Preset, this should be the Preset just created in this case. Otherwise, use the numeric keys or DATA slider to select the appropriate Preset.

6) Once the desired Preset is entered, use the DATA slider or the actual keyboard keys to select characters. Once a character is selected, use the right arrow, “>”, directly below the display window to select the next character. The left arrow, “<“, may be used to go back and modify previously selected characters as necessary. Available characters include:

DATA slider: ! “ # $ % & ’ ( ) * + , . / 0-9 : ; < = > ? @ A-Z [¥] ^ _ ‘ a-z { | } -> <- (94 characters)

Keyboard: ? @ A-Z [¥] ^ _ ‘ a-z { (61 characters)

<u>Notes</u>: - To select a blank space, slide the DATA slider to the bottom.
- Characters are listed in the order they appear on both the slider and physically on the keyboard.

7) When the display window shows the desired Preset (soon to be Bank) name, press **ENTER** [flashing].

8) The next step is to save the newly named Preset as a Bank on the hard disk. Change the drive by pressing: **DRIVE SELECT**, then enter **1**, the display will show:

SCSI1 : QUANTUM L
Avail: xxMB xx% **ENTER**

9) Now save the Preset just created to the desired Bank: select **PRESET MANAGEMENT**, option **8** [Save As 16 Bit Bank].

Save Bank to
Bxx NameYouPick

10) Select an unused Bank number (the display window should say "Bxx Empty Bank") with the DATA slider or the numeric keys. Press **ENTER** [flashing]. The display window will indicate:

Saving Bank...

If you desire to save to an already used Bank, it will be overwritten and will so indicate as a warning:

Save Bank to
Overwrites! OK?

To proceed the user must press the **ON /\ YES** button vice **ENTER.**

Note: If you have more than one Preset per Bank, the Bank will assume the name of the most recently entered Preset. Plan accordingly when setting up multiple Preset Banks. If Bank naming becomes a problem, just create a new Preset in the Bank in which you wish to change the name (**PRESET MANAGEMENT**, option **3**), give it the desired Bank name (**PRESET MANAGEMENT**, option **6**), and "Save As 16 Bit Bank" (**PRESET MANAGEMENT**, option **8**)

11) If the user wishes to abort at any point up to saving to the Bank, simply press **PRESET MANAGEMENT** once to cancel the operation. There is one exception to this rule of thumb, Rename Preset. Any changes made to the Preset name will be entered but not saved unless "Save As 16 Bit Bank" is performed. So if the name entered is not correct, simply Rename Preset (**PRESET MANAGEMENT**, option **6**), again.

C. TRANSFERRING FILES TO THE EMAX II FROM THE MACINTOSH USING SOUND DESIGNER II

Prior to transfer of files, the sound system should be set up in accordance with the Sound Sample Transfer Configuration in Appendix E (see page 61).

1) Energize the appropriate equipment:
   - Boot the Macintosh (if not already on)
   - Turn the Emax II power on (before the amplifier[1])
   - Turn the Carver amplifier power on (check volume levels)
   - Turn the Studio 3 MIDI Interface power on

1. It is important to apply power to the sampler prior to applying power to the amplifier to prevent damage to the speakers from spurious noise spikes often present during power on/off transitions.

2) When the Emax II is finished booting, load the desired bank and select the desired preset number.

3) On the Macintosh, start the Sound Designer II™ application from the desktop alias, the Sound Designer folder on the Zydaville disk, or by double clicking on any Sound Designer II file.

4) Once the soundfile to be transferred is open, select **SR Convert** from the DSP menu. For uniformity and optimum MIDI compatibility, set the **New Sample Rate** to 31,250 Hz and click on **Convert**. When asked to name the new file, use the original filename and add the character m as a suffix to indicate MIDI.

5) From the File menu, select **Save a Copy...** and Save using Sound Designer II Mono format. For continuity, files saved in the mono mode have the suffix m appended to the filename. Files saved with the MIDI suffix as well have the suffix mm.

6) Verify that the Emax II sampler has the desired bank and preset number selected for transfer and click on the Mac -> Sampler icon in the upper left corner of the display window. If the system is properly set up, when the icon is activated, the Studio 3 Modem MIDI In light and Channel 1, 2, and 3 lights should flash momentarily as the Macintosh verifies the communication path to the sampler.

7) Select **Add New Sample** or **Replace:** from the Transfer menu. The default preset will be from the bank initially selected in step 2 and verified in step 6. Use the arrow icons to step to the desired destination note on the sampler, when finished, click on **OK**. This action will cause the same lights to flash momentarily on Studio 3 as the Macintosh begins transferring data, and once again upon completion. The following advisory menu will appear on the Macintosh during transfer:

Sending sound data to the Emax II...
Samples remaining: xxx,xxx

At the same time the Emax II will indicate:

Receiving Data
Over RS-422...

8) Upon completion of transfer, test the sound by pressing the appropriate key on the Emax II. The lights on Studio 3 should flicker to indicate the flow of MIDI data. If the lights flicker and no sound is heard, check the volume levels on both the Emax II and the amplifier.

9) Once all of the desired files have been transferred, the preset must be saved before changing to another bank. If the new bank/preset is not saved, the next bank will be loaded into the sampler's RAM replacing the recently transferred (unsaved) bank. The transferred samples are lost and must be sent again. To save a preset as a 16 bit bank select **PRESET MANAGEMENT** option **8**. See steps 9 through 11 of "Creating a New Bank" on page 47 for further detail.

10) If the alert **There is a communication problem...** appears before or during the transfer, check the cabling set up. Ensure that the special DIN-8/DP-9 cable is connected and well seated on both ends and that the Modem and Printer MIDI/THRU switches are set to THRU on the Studio 3 Interface.

D. HEXADECIMAL VALUES FOR EMAX II KEYS

Various manuals, periodicals and other references provide note number equivalents for MIDI devices. When attempting to send hexadecimal MIDI commands to the Emax II from the IRIS sound server, it was determined that the values listed in the references consulted were inaccurate for the Emax II sound system. Following extensive testing, the values listed in Table 4 were recorded and confirmed to be accurate.

Appendix F contains a form for cataloging and recording Emax II preset data. The form allows the sound system administrator to maintain a list of sound file information for each preset within a bank. The form provides correlation between the Macintosh Sound Designer II sound name and the name used in simnet.h, as well as giving the key number and hexadecimal equivalent. Space for sample size (in bytes) and comments is provided also.

Table 4: HEXADECIMAL EQUIVALENTS OF EMAX II KEYS

Octave	C	C#	D	D#	E	F	F#	G	G#	A	A#	B
0	18	19	1A	1B	1C	1D	1E	1F	20	21	22	23
1	24	25	26	27	28	29	2A	2B	2C	2D	2E	2F
2	30	31	32	33	34	35	36	37	38	39	3A	3B
3	3C	3D	3E	3F	40	41	42	43	44	45	46	47
4	48	49	4A	4B	4C	4D	4E	4F	50	51	52	53
5	54	55	56	57	58	59	5A	5B	5C	5D	5E	5F
6	60	61	62	63	64	65	66	67	68	69	6A	6B

Note: Actual note range extends beyond the range of the physical keyboard. The physical keyboard extends from C1 to C6.

E. CURRENT EMAX II BANK PRESETS

The current sound file library implemented on NPSNET is indicated in Table 5. The data listed are in the format of the Preset Data Form of Appendix F.

Table 5: NPSNET CURRENT SOUND LIBRARY

Emax II note	Hex value	Sound Name	Sound.h name	Sample Size	Remarks
C3	3C	120MM Shot.mm	SHOT	63,641	
D3	3E	120MM Explosion.mm	EXPLOSION	129,626	
E3	40	Boom.mm	BOOM	149,984	
F3	41	ground_burst.mm	GROUND_BURST	46,347	
G3	43	rain.aiff.sd2.m	RAIN	190,976	Looped
A3	45	Bomber in flight.mm	BOMBER	219,136	Looped

APPENDIX D. SYSTEM PROCEDURES

A. SYSTEM SETUP

There are three primary configurations of the sound system that supports NPSNET. These configurations are listed in Appendix E along with their associated cabling requirements. This procedure describes the steps necessary to set up the peripheral devices which provide aural cues for normal operation of NPSNET. The procedures for recording sounds are located in Appendix B and the procedures for transferring sound files to the Emax II are found in Appendix C.

<u>General</u>

1) Prior to energizing any of the audio equipment, ensure the various devices are connected as indicated in Appendix E. Connecting electrical equipment with the power applied is not prudent, even though the majority of the device interconnections are digital or audio in nature.

<u>On the Emax II</u>

2) Turn the power switch located to the far left on the rear of the Emax II on. Once the sampler is finished booting from SCSI device 1 (the internal hard drive), select **LOAD BANK**, enter **34** with the numeric keypad or the DATA slider. The display should show:

Load Bank
B34 NPSNET sound   **ENTER** [flashing]

Press **ENTER**. The display window should indicate the bank is being loaded. Upon completion (approximately 6 seconds), the "NPSNET sound" bank will be ready for use.

<u>On the Carver Amplifier</u>

3) Turn the left and right volume levels to minimum, then press the power switch located to the far left of the face of the amplifier. Following a 2-3 second delay, the power on light should be illuminated, and the sampler volume level may be tested.

<u>On the Sound Server</u>

4) Login to the sound server (currently "elsie") and change to the NPSNET directory using the **dem** command (see Figure 4 on page 14).

5) Start NPSNET with the command **npsnet L T**, to activate the sound server and disable texturing on elsie. The **L** flag in the command line indicates the sound server option for a workstation (usually elsie), lower case **l** indicates a networked sound participant.

6) If difficulties are encountered when running NPSNET on the sound server, consult Chapter II's discussion of device port access (also Figure 7 on page 15).

On Other Graphics Workstations
7) Login and change directories as done in step 4.

8) Start NPSNET with the command **npsnet l** (texturing optional based on individual machine capability), to add a workstation to the simulation with networked aural cues.

Note: Both the **L** and **l** command line options automatically start NPSNET in the "networking on" mode.

B. INSTALLATION OF A NEW SOUND BYTE IN NPSNET

Creation of a new sound byte and inclusion in NPSNET is a fairly simple process that requires a few steps within the code of NPSNET and loading of the sound byte into the Emax II sound system.

On the Emax II
1) Follow the procedures outlined in Appendix C on page 49, for the transfer of a sound file to the Emax II from the Macintosh.

2) Consult Table 3 on page 52 for the appropriate hexadecimal value of the key to which the new sound byte is assigned.

In sound.h
3) Insert a #define for the new sound byte giving it a unique name and assign to it the hexadecimal value from step 2 above. See Figure 9 on page 23 for examples.

In the desired NPSNET file
4) Insert a call to the routine sendnetsoundmess() with the sound name from sound.h as the argument. For example: "sendnetsoundmess(BIG_BOOM);".

On the system
5) Recompile the NPSNET code using the make file to include the new sound byte.

6) Start NPSNET with one of the command line options given in Figure 4 on page 14. Ensure the system is set up as described in paragraph A above and in the Standard Operating Configuration described in Appendix E on page 56. Perform the action or initiate the desired event which generates the newly installed sound. Verify that the sound is generated in the desired manner. If the sound does not play, recheck the system set up and ensure that the proper bank and preset are loaded on the Emax II.

C. MOUNTING SYQUEST REMOVABLE DRIVES

1) To mount a different disk in a Syquest 44MB Removable drive, the Macintosh must be shut down, and one or both of the drives made available by removal of the currently mounted disk.

2) Once the new disk has been inserted and has come up to speed (indicated by the green ready light on the front of the drive), restart the Macintosh.

3) Select the Control Panel from the Apple menu and double-click on the SCSI Probe icon.

4) Select the appropriate drive ID (4 or 6) and choose "Mount" from the choices at the bottom of the window. If a warning window should come up saying "This disk is unrecognizable. Do you wish to initialize it?" Select "NO," unless it is a new or blank disk, and you do wish to initialize it. Selecting "Yes" erases the disk and reformats it, a highly undesirable result for a disk with data, sounds, or applications on it.

APPENDIX E. SYSTEM CONFIGURATIONS

A. PHYSICAL LAYOUT OF EQUIPMENT

The following figures give the general layout of the component devices that comprise the NPSNET sound system. Figure 13 shows the Macintosh side of the system, while Figure 14 gives an overhead view of the Emax II sound system, and a front view of the supporting equipment currently located behind the Emax II. For simplicity, the numerous cabling connections are not shown in these two figures.

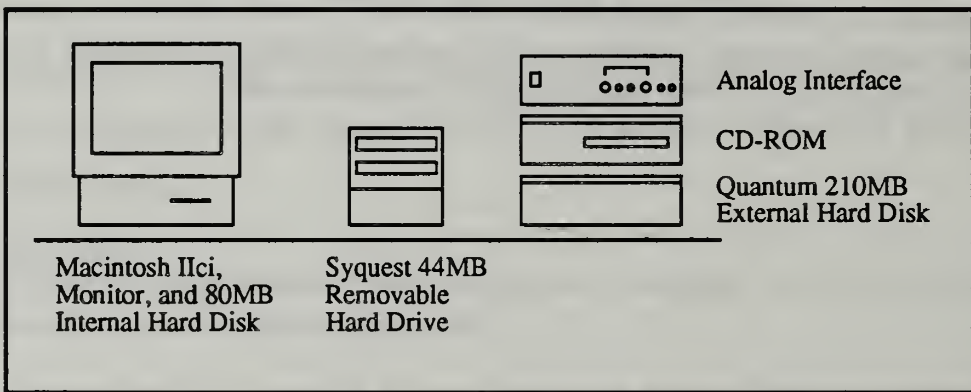

Figure 13 Physical Layout- Macintosh and Peripherals

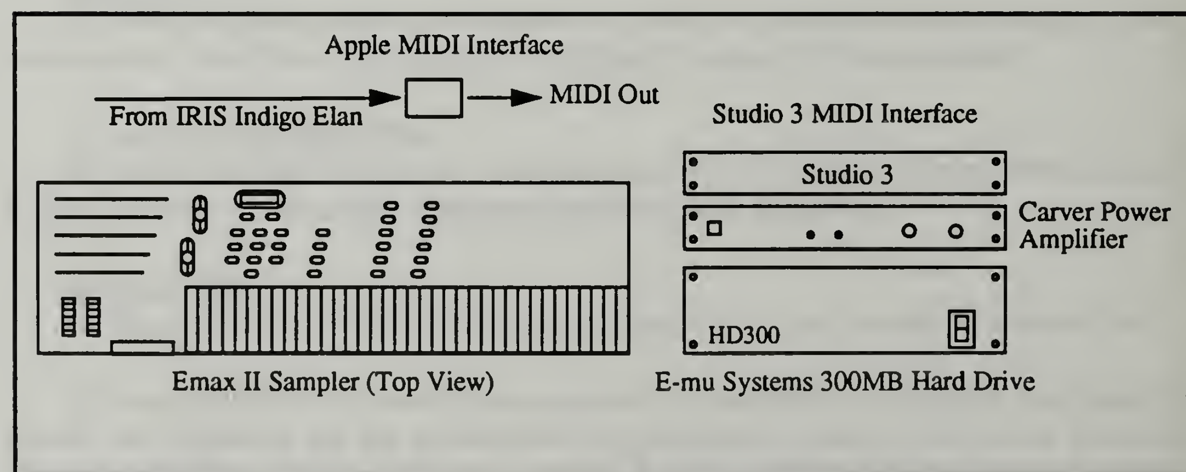

Figure 14 Physical Layout- Emax II and Peripherals

Note: The E-mu Systems 300MB Hard Drive, listed as a future expansion in the thesis body, arrived just prior to thesis completion and is included in this figure only.

B. COMPONENT REAR PANELS

The purpose of these rear panel illustrations (Figures 15, 16, and 17) is to provide a basis for the ensuing paragraphs which describe the various system configurations. The rear panel for the Studio 3 MIDI Interface may be found in Chapter II on page 11. A rear panel diagram for the IRIS workstation is not included, as only one connection is made between the IRIS and the Emax II via the Apple MIDI Interface.

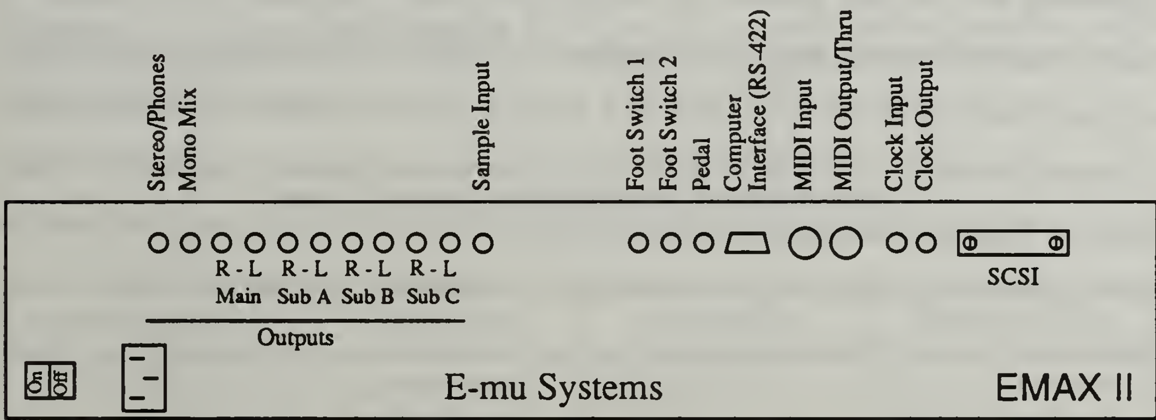

Figure 15 Emax II (Rear Panel- not to scale)

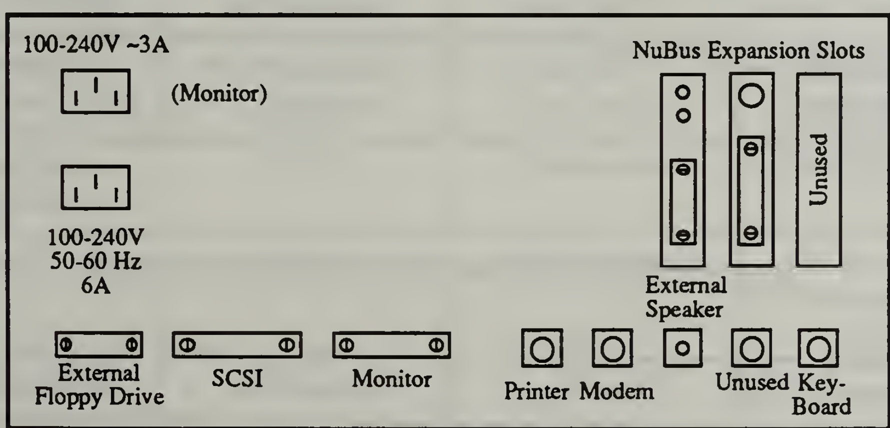

Figure 16 Macintosh IIci (Rear Panel)

Figure 17 Carver Power Amplifier (Rear Panel)

C. STANDARD OPERATING CONFIGURATION

In the standard operating configuration, the Emax II is the primary player in conjunction with the IRIS sound server. The Macintosh and its peripherals are not required to be in any specific setup for normal operation of NPSNET. Connections are listed below on an individual device basis in Tables 6, 7, 8, and 9. Since the Macintosh is not required for sound generation in NPSNET, Table 10 lists the normal operating connections which allow the Macintosh access to the local AppleTalk network. Basic connections such as power, monitor, and keyboard are assumed to be unchanging, and are not listed in these tables.

Table 6 IRIS INDIGO ELAN CONNECTIONS

Destination Device	Cable Type	Source Port	Destination Port
Apple MIDI Interface	DIN-8	#2 Serial Device port	Serial In

Table 7 APPLE MIDI INTERFACE CONNECTIONS

Destination Device	Cable Type	Source Port	Destination Port
Emax II	5-pin MIDI (black)	MIDI OUT	MIDI Input

Table 8 EMAX II SOUND SYSTEM CONNECTIONS

Destination Device	Cable Type	Source Port	Destination Port
Studio 3 Interface	DP-9 -> DIN-8	Computer Interface	Modem port/ modem icon
Carver Amplifier	Phono plug-> RCA plugs	Stereo/Phones	LINE In (red-right grey-left)

Note: The Studio 3 Interface connection is not mandatory for sound generation. However, it is recommended that the DP-9 end of the cable remain attached, while disconnecting the DIN-8 end because of the short cable length and to prevent wear and tear on the cable.

Table 9 CARVER AMPLIFIER CONNECTIONS

Destination Device	Cable Type	Source Port	Destination Port
Infinity Speakers	Copper wire, ends stripped	R+, R-, L+, L-	R+, R- Right spkr L+, L- Left spkr

Table 10 MACINTOSH NON-RECORDING CONNNECTIONS

Destination Device	Cable Type	Source Port	Destination Port
AppleTalk Network	DIN-8	Printer port	PhoneNET PLUS
Studio 3	DIN-8	Modem port	Modem port/ Computer icon
Syquest 44MB Drive	SCSI	SCSI	SCSI (4)

Note: The complete SCSI daisy chain is shown in Figure 1 on page 5, and should not be changed for any of the configurations in this appendix.

D. SOUND SAMPLE RECORDING CONFIGURATION

When recording sounds using the Macintosh, the Emax II sound system may be in any configuration. It is prudent however, to have the connections listed in Table 14 made to facilitate subsequent transfer of desired files.

Table 11 lists the port connections for stereo recording of sound files. To record in mono or in stereo using only one digitizer, connect the single digitizer to the modem port. The primary source of audio for recording with the Macintosh should be compact disc (CD) to ensure optimum sound quality. In this light, the connections for recording from CD-ROM are indicated in Table 12.

When using the mini plug line input from CD, only a single digitizer is required to receive both channels, providing the mini plug has two pickup rings. If a stereo mini plug is not available, the mini to dual RCA plug cable should be used with a single RCA to mini converter for each RCA plug. This connection is described in the second row of Table 12.

To record from another device, simply substitute that device name for the CD-ROM when making the appropriate connections. To record voice or from non-line sources, the digitizer connections are unnecessary, simply hold the microphone end of the digitizer near the source. Additional discussion of digitizer connection configurations may be found in [FARA 90] on pages 21-31.

Table 11 MACINTOSH RECORDING CONNECTIONS (STEREO)

Destination Device	Cable Type	Source Port	Destination Port
MacRecorder Digitizer	DIN-8 -> Digitizer	Modem port	Digitizer
MacRecorder Digitizer	DIN-8 -> Digitizer	Printer port	Digitizer

Table 12 CD-ROM CONNECTIONS

Destination Device	Cable Type	Source Port	Destination Port
MacRecorder Digitizer	Mini plug	Audio output	Digitizer line input
MacRecorder Digitizers	Mini plug -> stereo RCA -> Mini plug(2)	Audio output	Digitizer line input via adapters

E. SOUND SAMPLE TRANSFER CONFIGURATION

To minimize cable swapping, two sets of MIDI cables are used, one set to interface between the Studio 3 Interface (red and black) and the second set to interface with the Apple MIDI Interface (grey, A and B). When recording and sending samples to the Emax II from the Macintosh, the grey MIDI In and Out/Thru cables should be used. These cables should be left attached to the back of the Studio 3 Interface and connected to the Emax II sound system when sending samples from the Macintosh.

Table 13 MACINTOSH SOUND TRANSFER CONNECTIONS

Destination Device	Cable Type	Source Port	Destination Port
Studio 3	DIN-8	Modem port	Modem port/ Computer icon
Studio 3	DIN-8	Printer port	Printer port/ Computer icon

<u>Note</u>: The Printer port connection is not required in this configuration, it is simply listed to complete logical data flow. The Emax II protocol sends acknowledgments across the same cable that the sound samples are transferred.

Table 14 EMAX II SOUND TRANSFER CONNECTIONS

Destination Device	Cable Type	Source Port	Destination Port
Studio 3	DP-9 -> DIN-8	Computer Interface	Modem port/ modem icon
Studio 3	5-pin MIDI (A)	MIDI Input	MIDI Out (ch 1)
Studio 3	5-pin MIDI (B)	MIDI Output/Thru	MIDI In/ modem icon

Note: Any of the 6 MIDI output channels may be used, channel 1 is used for consistency. The MIDI connections are primarily used for monitoring of data flow.

APPENDIX F. EMAX II PRESET DATA FORM

Bank ____ Preset____________________________

Note	Hex	Sound Name	Sound.h name	Sample size	Comments
C1	24				
D1	26				
E1	28				
F1	29				
G1	2B				
A1	2D				
B1	2F				
C2	30				
D2	32				
E2	34				
F2	35				
G2	37				
A2	39				
B2	3B				
C3	3C				
D3	3E				
E3	40				
F3	41				
G3	43				
A3	45				
B3	47				
C4	48				
D4	4A				
E4	4C				
F4	4D				
G4	4F				
A4	51				
B4	53				
C5	54				
D5	56				
E5	58				
F5	59				
G5	5B				
A5	5D				
B5	5F				
C6	60				

LIST OF REFERENCES

[ALTE 86] Alten, Stanley R., *Audio in Media*, Wadsworth Publishing Company, 1986.

[ANDE 86] Anderton, Craig, *MIDI for Musicians*, Amsco Publications, 1986.

[BACH 86] Bach, Maurice J., *The Design of the Unix Operating System*, Prentice-Hall, Inc., 1986.

[BEGA 90] Begault, Durand R. and Wenzel, Elisabeth M. *Techniques and Applications for Binaural Sound Manipulation in Human-Machine Interfaces*, NASA Internal Tech Report, August 1990.

[BENE 92] Benedikt, Michael, Ed., *Cyberspace: First Steps*, Massachusetts Institute of Technology, 1992.

[BRUT 92] Brutzman, Donald P., *NPS AUV Integrated Simulator*, M.S. Thesis, Naval Postgraduate School, Monterey, California, March 1992.

[DIGI 90] Digidesign Incorporated, *Sound Designer II, Version 2.0*, Digidesign Incorporated, 1990.

[E-MU 89] E-mu Systems, Incorporated, *Emax II 16-Bit Digital Sound System Operation Manual*, E-mu Systems, Incorporated, 1989.

[FARA 90] Farallon Computing, Incorporated, *MacRecorder Sound System*, Farallon Computing Incorporated, 1990.

[FRIE 92] Friedmann, Martin, Starner, Thad and Pentland, Alex, "Device Synchronization Using an Optimal Linear Filter," Computer Grahpics, 1992 Symposium on Interactive 3D Graphics, March 1992, pp. 57.

[GIBS 84] Gibson, William, *Neuromancer*, Ace Books, 1984.

[GIBS 86] Gibson, William, *Count Zero*, Grafton Books, 1986.

[HUBE 90] Huber, David Miles, *Random Access Audio*, Digidesign, Incorporated, 1990.

[HUBE 91] Huber, David Miles, *The MIDI Manual*, SAMS Division of Macmillan Computer Publishing, 1991.

[KANG 92] Kang, George S., and Heide, David A., "Canned Speech for Tactical Voice Message Systems," Presentation from the 1992 Tactical Communications Conference, Fort Wayne, Indiana, April 28-30, 1992.

[LAUR 91] Laurel, Brenda, "Be There Here," on ACM SIGGRAPH CHI '92 Special Video Program, Issue 78, No.6, 1991, entitled "Future Scenarios: Virtual Reality".

[OPCO 90a] Opcode Systems, Incorporated, *Studio 3 User's Guide*, Opcode Systems, Incorporated, 1990.

[OPCO 90b] Opcode Systems, Incorporated, *Studio Vision Integrated MIDI and Digital Audio Recording, Version 1.3*, Opcode Systems Incorporated, 1990.

[POHL 85] Pohlman, Ken C., *Principles of Digital Audio*, Howard W. Sams and Company, 1985.

[RHEI 91] Rheingold, Howard, *Virtual Reality*, Summit Press, 1991.

[SUTH 68] Sutherland, I. E., "A Head-Mounted Three Dimensional Display," Fall Joint Computer Conference (FJCC), Thompson Books, 1968, pp. 757-764.

[TAKA 92] Takala, Tapio and Hahn, James, "Sound Rendering," SIGGRAPH '92, ACM Computer Graphics, Vol. 26, No. 2, pp. 211-220.

[WENZ 90] Wenzel, Elizabeth M. and Foster, Scott H., "Realtime Digital Synthesis of Virtual Acoustic Environments," Copmuter Graphics, Vol. 24, No. 2, March 1990, pp. 139.

[WENZ 92] Wenzel, Elizabeth M. "Localization in Virtual Acoustic Displays," Presence, Vol. 1, No. 1, Winter 1992, pp. 80.

[YELT 89] Yelton, Geary, *Music and the Macintosh*, MIDI America, 1989.

[ZYDA 92] Zyda, Michael J., Pratt, David R., Monahan, James G. and Wilson, Kalin P., "NPSNET: Constructing a 3D Virtual World," in *Computer Graphics, Special Issue on the 1992 Symposium on Interactive 3D Graphics*, MIT Media Laboratory, 29 March - 1 April 1992, pp. 147-156.

INITIAL DISTRIBUTION LIST

1.	Defense Technical Information Center Cameron Station Alexandria, VA 22304-6145	2
2.	Dudley Knox Library Code 52 Naval Postgraduate School Monterey, CA 93943-5002	2
3.	Dr. Michael J. Zyda Code CS/ZK, Department of Computer Science Naval Postgraduate School Monterey, CA 93943-5100	4
4.	David R. Pratt Code CS/Pr, Department of Computer Science Naval Postgraduate School Monterey, CA 93943-5100	3
5.	LCDR Donald P. Brutzman Code OR/BR, Department of Operations Research Naval Postgraduate School Monterey, CA 93943-5100	1
6.	Lieutenant Leif A. Dahl 11919 Old Pomerado Road Poway, CA 92064	1
7.	Dr. Susannah Bloch Platt Campus Center Harvey Mudd College Claremont, CA 91711	1
8.	Phil Faris LE Centre, Multilingual Multimedia Educational Software 220 D San Benancio Canyon Salinas, CA 93908	1

Thesis
D127 Dahl
c.1 NPSNET.